



# PCI 6540 Data Book

---





# PCI 6540 Data Book

---

Version 1.0

November 2003

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

© 2003 PLX Technology, Inc. All rights reserved.

PLX Technology, Inc., retains the right to make changes to this product at any time, without notice. Products may have minor variations to this publication, known as errata. PLX assumes no liability whatsoever, including infringement of any patent or copyright, for sale and use of PLX products.

This device is not designed, intended, authorized, or warranted to be suitable for use in medical or life-support applications, devices, or systems, or other critical applications.

PLX Technology and the PLX logo are registered trademarks of PLX Technology, Inc.

HyperTransport is a trademark of the HyperTransport Technology Consortium.

Other brands and names are the property of their respective owners.

Order Number: PCI 6540-SIL-DB-P1-1.0

Printed in the USA, November 2003

# Contents

<b>Figures</b> .....	<b>xiii</b>
<b>Tables</b> .....	<b>xv</b>
<b>Registers</b> .....	<b>xvii</b>
<b>Preface</b> .....	<b>xxiii</b>
Supplemental Documentation .....	xxiii
Data Assignment Conventions .....	xxiv
Revision History .....	xxiv
<b>Feature Summary</b> .....	<b>xxv</b>
<b>1. Introduction</b> .....	<b>1-1</b>
1.1. Company and Product Information .....	1-1
1.2. PCI 6000 Bridge Series .....	1-2
1.2.1. PCI 6540 .....	1-3
1.3. Feature Description .....	1-4
1.4. Applications .....	1-4
1.4.1. Multiple Device Expansion .....	1-4
1.4.2. Intelligent Adapters .....	1-5
1.4.3. CompactPCI Universal Application .....	1-5
<b>2. Functional Overview</b> .....	<b>2-1</b>
2.1. General Operation .....	2-1
2.2. Write Transactions .....	2-2
2.3. Read Transactions .....	2-2
2.4. Non-Transparent Mode .....	2-2
<b>3. Pin Description</b> .....	<b>3-1</b>
3.1. Pin Summary .....	3-1
3.2. Pull-Up and Pull-Down Resistor Recommendations .....	3-1
3.2.1. Primary and Secondary PCI Bus Interface Pins .....	3-1
3.2.2. Primary and Secondary Clock Pins .....	3-2
3.2.3. Reset Pins .....	3-2
3.2.4. CompactPCI Hot Swap Pins .....	3-2
3.2.5. JTAG/Boundary Scan Pins .....	3-2
3.2.6. Serial EEPROM Pins .....	3-3
3.2.7. GPIO Pins .....	3-3
3.2.8. Miscellaneous Pins .....	3-3
3.2.9. Multiplexed Transparent and Non-Transparent Pins .....	3-3
3.3. Power Supply Decoupling .....	3-3
3.4. Pinout Common to All Operating Modes .....	3-4
3.5. Pinout Specific to Transparent and Non-Transparent Modes .....	3-25

<b>4. Clocking</b>	<b>4-1</b>
4.1. Primary and Secondary Clock Inputs	4-1
4.2. Secondary Clock Outputs	4-1
4.3. Disabling Secondary Clock Outputs	4-1
4.4. Using an External Clock Source	4-1
4.5. Frequency Division options	4-2
4.6. Running Secondary Port Faster than Primary Port	4-2
4.7. Universal Mode Clock Behavior	4-2
4.8. PLL and Clock Jitter	4-3
4.9. Detecting PCI Bus Speed with the Reference Clock	4-4
4.10. Primary or Secondary Clock Frequency Measurement	4-4
<b>5. Reset and Initialization</b>	<b>5-1</b>
5.1. PCI-XCAP Connections and Operating Frequency	5-1
5.1.1. Primary Port PCI-XCAP Connection	5-1
5.1.2. Secondary Port PCI-XCAP Connection	5-1
5.2. Secondary Bus Mode and Frequency Initialization Sequence	5-1
5.3. Conventional PCI Mode 66 MHZ Operation	5-2
5.4. Reset	5-2
5.4.1. Power Good Reset	5-2
5.4.1.1. PWRGD, Resets, and PCI-X Signals Relationship	5-4
5.4.2. Primary Reset Input	5-6
5.4.3. Primary Reset Output	5-6
5.4.4. Secondary Reset Input	5-6
5.4.4.1. Universal Mode Secondary Reset Input	5-7
5.4.5. Secondary Reset Output	5-7
5.4.6. Software Resets	5-7
5.4.7. Power Management Internal Reset	5-7
5.4.8. Reset Inputs Table	5-8
5.4.9. Power-Up and Reset Pin State Table	5-9
5.5. Register Initialization	5-14
5.5.1. Default Initialization	5-14
5.5.2. Serial EEPROM Initialization	5-14
5.5.3. Host Initialization	5-14
<b>6. Registers</b>	<b>6-1</b>
6.1. PCI Configuration Register Address Mapping—Transparent Mode	6-2
6.1.1. PCI Type 1 Header Registers—Transparent Mode	6-4
6.1.2. Device-Specific Registers—Transparent Mode	6-16
6.1.2.1. Chip, Diagnostic, and Arbiter Control Registers	6-16
6.1.2.2. Primary Flow-Through Control Register	6-18
6.1.2.3. Timeout Control Register	6-19
6.1.2.4. Miscellaneous Options Register	6-20
6.1.2.5. Prefetch Control Registers	6-22
6.1.2.6. Secondary Flow-Through Control Register	6-26
6.1.2.7. Buffer and Internal Arbiter Control Registers	6-27
6.1.2.8. Test and Serial EEPROM Registers	6-29
6.1.2.9. Timer Registers	6-31
6.1.2.10. Primary System Error Event Register	6-32
6.1.2.11. GPIO[3:0] Registers	6-33
6.1.2.12. Clock Control Register	6-34
6.1.2.13. Primary System Error Status Register	6-34
6.1.2.14. Clock Run Register	6-35
6.1.2.15. Private Memory Registers	6-36
6.1.2.16. Hot Swap and Read-Only Register Control Register	6-37
6.1.2.17. GPIO[7:4], Power-Up Status, and GPIO[15:8] Registers	6-38

6.1.2.18. Sticky Scratch and Extended Registers . . . . .	6-40
6.1.2.19. Address Translation Control Registers . . . . .	6-42
6.1.2.20. Power Management Capability Registers . . . . .	6-51
6.1.2.21. Hot Swap Capability Registers . . . . .	6-54
6.1.2.22. VPD Capability Registers . . . . .	6-55
6.1.2.23. PCI-X Capability Registers . . . . .	6-56
6.2. PCI Configuration Register Address Mapping—Non-Transparent Mode . . . . .	6-60
6.2.1. PCI Configuration Register Address Mapping 00h to 7Fh—Non-Transparent Mode . . . . .	6-60
6.2.2. Primary Configuration Registers—Non-Transparent Mode . . . . .	6-62
6.2.2.1. Primary Port PCI Type 0 Header Registers . . . . .	6-62
6.2.2.2. Secondary Port PCI Type 0 Header Registers . . . . .	6-72
6.2.3. PCI Shadow Configuration Registers—Non-Transparent Mode . . . . .	6-81
6.2.3.1. Primary Flow-Through Control Registers . . . . .	6-82
6.2.3.2. Timeout Control Register . . . . .	6-85
6.2.3.3. Miscellaneous Options Register . . . . .	6-86
6.2.3.4. Prefetch Control Registers . . . . .	6-88
6.2.3.5. Secondary Flow-Through Control Register . . . . .	6-92
6.2.3.6. Buffer and Internal Arbiter Control Registers . . . . .	6-93
6.2.3.7. Test and Serial EEPROM Registers . . . . .	6-95
6.2.3.8. Timer Registers . . . . .	6-97
6.2.4. PCI Configuration Register Address Mapping 80h to FFh, Shadow and Extended Registers— Non-Transparent Mode . . . . .	6-98
6.2.4.1. Configuration Registers 80h to FFh . . . . .	6-98
6.2.4.2. Cross-Bridge Configuration Access Control Registers . . . . .	6-100
6.2.4.3. Clock Control Register . . . . .	6-102
6.2.4.4. System Error Event Registers . . . . .	6-103
6.2.4.5. GPIO[3:0] Registers . . . . .	6-106
6.2.4.6. Hot Swap and Read-Only Control Register . . . . .	6-107
6.2.4.7. GPIO[7:4], Power-Up Status, and GPIO[15:8] Registers . . . . .	6-108
6.2.4.8. Direct Message Interrupt Registers . . . . .	6-110
6.2.4.9. Message Signaled Interrupt Registers . . . . .	6-112
6.2.4.10. Doorbell and Miscellaneous Interrupt Registers . . . . .	6-114
6.2.4.11. Non-Transparent Configuration Ownership Semaphore Register . . . . .	6-117
6.2.4.12. Sticky Scratch and Extended Registers . . . . .	6-118
6.2.4.13. Address Translation Control Registers . . . . .	6-120
6.2.4.14. Chip, Diagnostic, and Arbiter Control Registers . . . . .	6-127
6.2.4.15. Power Management Capability Registers . . . . .	6-129
6.2.4.16. Hot Swap Capability Registers . . . . .	6-132
6.2.4.17. VPD Capability Registers . . . . .	6-133
6.2.4.18. PCI-X Capability Registers . . . . .	6-134
<b>7. Serial EEPROM . . . . .</b>	<b>7-1</b>
7.1. Overview . . . . .	7-1
7.2. Serial EEPROM Access . . . . .	7-1
7.3. Serial EEPROM Autoload Mode . . . . .	7-1
7.4. Universal Non-Transparent Mode Groups . . . . .	7-2
7.5. Serial EEPROM Data Structure . . . . .	7-2
7.5.1. Serial EEPROM Address and Corresponding PCI 6540 Register . . . . .	7-3
<b>8. PCI Bus Operation . . . . .</b>	<b>8-1</b>
8.1. Conventional PCI Transactions . . . . .	8-1
8.2. Single Address Phase . . . . .	8-2
8.3. Dual Address Phase . . . . .	8-2
8.4. Device Select (DEVSEL#) Generation . . . . .	8-2
8.5. Data Phase . . . . .	8-2
8.5.1. Posted Write Transactions . . . . .	8-3
8.5.2. Memory Write and Invalidate Transactions . . . . .	8-3

8.5.3. Delayed Write Transactions . . . . .	8-4
8.5.4. Write Transaction Address Boundaries . . . . .	8-5
8.5.5. Buffering Multiple Write Transactions . . . . .	8-5
8.5.6. Read Transactions . . . . .	8-5
8.5.7. Prefetchable Read Transactions . . . . .	8-6
8.5.8. Non-Prefetchable Read Transactions . . . . .	8-6
8.5.9. Read Prefetch Address Boundaries . . . . .	8-6
8.5.10. Delayed Read Requests . . . . .	8-7
8.5.11. Delayed Read Completion with Target . . . . .	8-7
8.5.12. Delayed Read Completion on Initiator Bus . . . . .	8-7
8.5.13. Configuration Transactions . . . . .	8-8
8.5.14. PCI 6540 Type 0 Access . . . . .	8-8
8.5.15. Type 1-to-Type 0 Translation . . . . .	8-9
8.5.16. Type 1-to-Type 1 Forwarding . . . . .	8-11
8.5.17. Special Cycles . . . . .	8-11
8.6. Transaction Termination in Conventional PCI Mode . . . . .	8-12
8.6.1. PCI 6540-Initiated Master Termination . . . . .	8-12
8.6.2. Master Abort Received by PCI 6540 . . . . .	8-12
8.6.3. Target Termination Received by PCI 6540 . . . . .	8-13
8.6.3.1. Delayed Write Target Termination Response . . . . .	8-14
8.6.3.2. Posted Write Target Termination Response . . . . .	8-15
8.6.3.3. Delayed Read Target Termination Response . . . . .	8-16
8.6.4. PCI 6540-Initiated Target Termination . . . . .	8-17
8.6.4.1. Target Retry . . . . .	8-17
8.6.4.2. Target Disconnect . . . . .	8-18
8.6.4.3. Target Abort . . . . .	8-18

## 9. PCI-X Bus Operation . . . . . 9-1

9.1. Overview . . . . .	9-1
9.2. General Bus Rules . . . . .	9-1
9.2.1. Initiator Rules . . . . .	9-2
9.2.2. Target Rules . . . . .	9-2
9.2.3. Bus Arbitration Rules . . . . .	9-3
9.2.4. Configuration Transaction Rules . . . . .	9-3
9.2.5. Parity Error Rules . . . . .	9-3
9.2.6. Bus Data Width Rules . . . . .	9-3
9.2.7. Split Transaction Rules . . . . .	9-3
9.3. PCI-X Sequences . . . . .	9-4
9.4. ADB and Buffer Size . . . . .	9-4
9.5. Dependencies between AD and CBE# . . . . .	9-4
9.6. PCI-X Command Encoding . . . . .	9-5
9.7. Attributes . . . . .	9-6
9.8. Burst Transactions . . . . .	9-8
9.8.1. Burst Write and Split Completion . . . . .	9-8
9.8.2. Burst Read Transactions . . . . .	9-8
9.9. DWORD Transactions . . . . .	9-8
9.10. Device Select Timing . . . . .	9-9
9.11. Wait States and Target Initial Latency . . . . .	9-9
9.12. Split Transactions . . . . .	9-10
9.12.1. Split Completion Transaction . . . . .	9-11
9.12.2. Immediate Completion by the Completer . . . . .	9-11
9.12.3. Split Completion Address . . . . .	9-11
9.12.4. Completer Attributes . . . . .	9-13
9.12.5. Accepting Split Completions Requirement . . . . .	9-14
9.12.6. Split Completion Messages . . . . .	9-14



9.13. PCI-X Mode Transaction Termination . . . . .	9-17
9.13.1. PCI 6540 Initiator Termination . . . . .	9-17
9.13.1.1. Byte Count Disconnection or Satisfaction . . . . .	9-17
9.13.1.2. PCI 6540 Master Abort Termination . . . . .	9-17
9.13.2. PCI 6540 Target Termination . . . . .	9-17
9.13.2.1. PCI 6540 Disconnects at Next ADB . . . . .	9-19
9.13.2.2. PCI 6540 Retry Termination . . . . .	9-19
9.13.2.3. PCI 6540 Split Response Termination . . . . .	9-19
9.14. PCI-X Mode Bus and Data Transfer Width . . . . .	9-19
9.15. Connecting Conventional PCI and PCI-X Interfaces . . . . .	9-20
9.15.1. Conventional PCI Requester, PCI-X Completer . . . . .	9-20
9.15.2. PCI-X Requester, Conventional PCI Completer . . . . .	9-21
<b>10. Address Decoding . . . . .</b>	<b>10-1</b>
10.1. Overview . . . . .	10-1
10.2. Address Ranges . . . . .	10-1
10.2.1. I/O Address Decoding . . . . .	10-1
10.2.1.1. I/O Base and Limit Address Registers . . . . .	10-1
10.3. Memory Address Decoding . . . . .	10-2
10.3.1. Memory-Mapped I/O Base and Limit Address Registers . . . . .	10-3
10.3.1.1. Prefetchable Memory Base and Limit Address Registers . . . . .	10-3
10.4. ISA Mode . . . . .	10-4
10.5. VGA Support . . . . .	10-4
10.5.1. VGA Mode . . . . .	10-4
10.5.2. VGA Snoop Mode . . . . .	10-5
10.6. Private Device Support . . . . .	10-5
10.7. Address Translation . . . . .	10-5
10.7.1. Base Address Registers . . . . .	10-5
10.7.2. Transparent Mode Address Translation . . . . .	10-6
10.7.2.1. Transparent Mode Address Translation Method . . . . .	10-6
10.7.2.2. Register Access . . . . .	10-6
10.7.2.3. Address Translation on Primary-to-Secondary (Downstream) Transactions . . . . .	10-9
10.7.2.4. Address Translation on Secondary-to-Primary (Upstream) Transactions . . . . .	10-10
10.7.2.5. Serial EEPROM Configuration of Transparent Mode Address Translation . . . . .	10-11
10.7.3. Non-Transparent Mode Address Translation . . . . .	10-11
<b>11. Transaction Ordering . . . . .</b>	<b>11-1</b>
11.1. Conventional PCI Transaction Ordering . . . . .	11-1
11.1.1. Transactions Governed by Ordering Rules . . . . .	11-1
11.1.2. General Ordering Guidelines . . . . .	11-1
11.1.3. Ordering Rules . . . . .	11-2
11.1.4. Data Synchronization . . . . .	11-4
11.2. PCI-X Transaction Ordering . . . . .	11-4
11.2.1. Relaxed Ordering Attribute Bit . . . . .	11-4
11.2.2. Split Transactions . . . . .	11-4
<b>12. Error Handling . . . . .</b>	<b>12-1</b>
12.1. Overview . . . . .	12-1
12.2. Address Parity Errors . . . . .	12-1
12.3. Attribute Parity Errors (PCI-X Mode) . . . . .	12-2
12.4. Data Parity Errors . . . . .	12-2
12.4.1. Configuration Write Transactions to Configuration Space . . . . .	12-2
12.4.2. Read Transactions . . . . .	12-2
12.4.3. Delayed Write Transactions . . . . .	12-3
12.4.3.1. Conventional PCI Mode . . . . .	12-3
12.4.3.2. PCI-X Mode . . . . .	12-3

12.4.4. Split Completion (PCI-X Mode) . . . . .	12-4
12.4.5. Posted Write Transactions . . . . .	12-4
12.5. Data Parity Error Reporting Summary . . . . .	12-5
12.6. System Error (SERR#) Reporting . . . . .	12-6
<b>13. Exclusive Access . . . . .</b>	<b>13-1</b>
13.1. Concurrent Locks . . . . .	13-1
13.2. Acquiring Exclusive Access across PCI 6540 . . . . .	13-1
13.3. Ending Exclusive Access . . . . .	13-2
<b>14. PCI Bus Arbitration . . . . .</b>	<b>14-1</b>
14.1. Overview . . . . .	14-1
14.2. Primary PCI Bus Arbitration . . . . .	14-1
14.3. Secondary PCI Bus Arbitration . . . . .	14-1
14.3.1. Secondary Bus Arbitration Using Internal Arbiter . . . . .	14-1
14.3.2. Rotating Priority Scheme . . . . .	14-2
14.3.3. Fixed Priority Scheme . . . . .	14-2
14.3.4. Secondary Bus Arbitration Using External Arbiter . . . . .	14-3
14.4. Arbitration Bus Parking . . . . .	14-3
14.4.1. Software Controlled PCI 64-Bit Extension Signals Parking . . . . .	14-3
<b>15. GPIO Interface . . . . .</b>	<b>15-1</b>
15.1. GPIO Interface Pins . . . . .	15-1
15.2. GPIO Control Registers . . . . .	15-1
<b>16. Supported Commands . . . . .</b>	<b>16-1</b>
16.1. Primary Interface Command Set . . . . .	16-1
16.2. Secondary Interface Command Set . . . . .	16-3
<b>17. Bridge Behavior . . . . .</b>	<b>17-1</b>
17.1. Bridge Actions for Various Cycle Types . . . . .	17-1
17.2. Abnormal Termination (Initiated by Bridge Master) . . . . .	17-2
17.2.1. Master Abort . . . . .	17-2
17.3. Parity and Error Reporting . . . . .	17-2
17.3.1. Reporting Parity Errors . . . . .	17-2
17.4. S_IDSEL Mapping . . . . .	17-2
17.5. 32- to 64-bit Cycle Conversion . . . . .	17-3
<b>18. PCI Flow-Through Optimization . . . . .</b>	<b>18-1</b>
18.1. Overview . . . . .	18-1
18.2. Precautions when Using Non-Optimized PCI Master Devices . . . . .	18-1
18.3. Posted Write Flow Through . . . . .	18-1
18.4. Delayed Read Flow Through . . . . .	18-2
18.5. Read Cycle Optimization . . . . .	18-2
18.5.1. Primary and Secondary Initial Prefetch Count . . . . .	18-2
18.5.2. Primary and Secondary Incremental Prefetch Count . . . . .	18-3
18.5.3. Primary and Secondary Maximum Prefetch Count . . . . .	18-3
18.6. Read Prefetch Boundaries . . . . .	18-3

<b>19. FIFO Architecture</b>	<b>19-1</b>
19.1. Overview	19-1
19.2. Memory Writes	19-2
19.2.1. PCI-to-PCI-X Memory Writes	19-2
19.2.2. PCI-X-to-PCI Memory Writes	19-2
19.2.3. PCI-X-to-PCI-X Memory Writes	19-2
19.3. Memory Reads	19-2
19.3.1. PCI-to-PCI-X Memory Reads	19-2
19.3.1.1. Prefetched Data Timeout Flushing	19-3
19.3.1.2. Setting the Prefetch Count	19-4
19.3.1.2.1. PCI Read from PCI-X Port	19-4
19.3.1.2.2. PCI Read from PCI Port	19-4
19.3.2. PCI-X-to-PCI Memory Reads	19-4
19.3.3. PCI-X-to-PCI-X Memory Reads	19-4
19.4. Splitting the Read FIFO into Four 1-KB Blocks—Transparent Mode	19-4
<b>20. Non-Transparent Mode</b>	<b>20-1</b>
20.1. Overview	20-1
20.2. Using XB_MEM Input to Avoid Initial Retry Latency	20-2
20.3. Interrupts	20-2
20.3.1. Direct Message Interrupts	20-2
20.3.2. Doorbell Interrupts	20-2
20.3.3. Message Signaled Interrupts	20-2
20.4. Non-Transparent Mode Power-Up/PCI Reset Sequence	20-2
<b>21. Power Management</b>	<b>21-1</b>
21.1. Overview	21-1
21.2. P_PME# and S_PME# Signals	21-1
<b>22. Hot Swap</b>	<b>22-1</b>
22.1. Overview	22-1
22.2. Early Power Support	22-1
22.3. Hot Swap Signals	22-1
22.4. Hot Swap Register Control and Status	22-2
22.5. Avoiding Initially Retry or Initially Not Responding Requirement	22-2
22.6. Device Hiding	22-2
22.7. Implementing Hot Swap Controller Using PCI 6540 GPIO Pins	22-2
22.8. Running PCI-X in Hot Swap Slots	22-2
<b>23. VPD</b>	<b>23-1</b>
<b>24. Testability/Debug</b>	<b>24-1</b>
24.1. JTAG Interface	24-1
24.1.1. IEEE 1149.1 Test Access Port	24-1
24.1.2. JTAG Instructions	24-1
24.1.3. JTAG Boundary Scan	24-2
24.1.4. JTAG Reset Input TRST#	24-2
<b>25. Electrical Specs</b>	<b>25-1</b>
25.1. General Electrical Specifications	25-1
25.2. PLL and Clock Jitter	25-3
25.3. PCI-X/PCI Signal Timing Specification	25-4

<b>26. Mechanical Specs</b> .....	<b>26-1</b>
26.1. Mechanical Dimensions .....	26-1
26.2. Pinout .....	26-4
<b>A. Using PCI 6540.</b> .....	<b>A-1</b>
A.1. Transparent Mode Application .....	A-2
A.2. Non-Transparent Mode Application .....	A-3
A.3. Universal Bridging Application .....	A-4
A.3.1. Universal Mode CLK, RST#, REQ0#, GNT0#, and SYSEN# Signal Connections .....	A-5
A.4. Symmetrical Non-Transparent Application .....	A-7
<b>B. PCI-X Clock and Frequency Initialization Sequence</b> .....	<b>B-1</b>
B.1. Bus Speed and Type Detection .....	B-1
B.1.1. Universal Transparent Mode .....	B-2
B.1.2. Universal Non-Transparent Mode .....	B-2
B.2. Secondary Clock Outputs .....	B-2
B.3. Internal Clock Divider .....	B-2
B.4. PCI-X in Universal Mode S_CLKO0 Pin Use .....	B-2
<b>C. PCI 6520BB and PCI 6540BB Pin Comparison</b> .....	<b>C-1</b>
C.1. Overview .....	C-1
<b>D. General Information</b> .....	<b>D-1</b>
D.1. Ordering .....	D-1
D.2. United States and International Representatives, and Distributors .....	D-2
D.3. Technical Support .....	D-2
<b>Index</b> .....	<b>Index-1</b>

# FIGURES

1-1. PCI 6000 Series of PCI-to-PCI Bridges	1-2
1-2. PCI 6540 PCI-X-to-PCI-X Bridge	1-3
1-3. Multiple Device Expansion	1-4
1-4. Intelligent Adapters	1-5
1-5. CompactPCI Universal Application	1-5
2-1. PCI 6540 Block Diagram	2-1
7-1. Serial EEPROM Data Structure	7-2
9-1. Requester Attribute Bit Assignments	9-7
9-2. Split Completion Address	9-12
9-3. Completer Attribute Bit Assignments	9-13
9-4. Split Completion Message Attribute Bit Assignments	9-15
14-1. Secondary Bus Arbiter Example	14-2
19-1. FIFO Architecture	19-1
20-1. Non-Transparent Mode Power-Up/PCI Reset Sequence	20-3
22-1. Hot Insertion Power-Up Sequence Recommendation	22-1
25-1. PCI-X/PCI Signal Timing Specification	25-4
26-1. PCI 6540 Mechanical Dimensions	26-1
26-2. PCI 6540 Physical Layout with Pinout—Topside View (A1–A10 through Y1–Y10)	26-4
26-3. PCI 6540 Physical Layout with Pinout—Topside View (A11–A20 through Y11–Y20)	26-5



# TABLES

2-1. Non-Transparent Mode Base Address Registers .....	2-2
3-1. Pin Type Abbreviations .....	3-1
3-2. Generic PCI Bus Interface Pins that follow <i>PCI r2.3</i> and <i>PCI-X r1.0b</i> Layout Guidelines .....	3-1
3-3. Clock Pin Pull-Up/Pull-Down Resistor Requirements .....	3-2
3-4. Primary PCI/PCI-X Bus Interface Pins .....	3-4
3-5. Secondary PCI/PCI-X Bus Interface Pins .....	3-8
3-6. Clock-Related Pins .....	3-13
3-7. Reset Pins .....	3-16
3-8. CompactPCI Hot Swap Pins .....	3-17
3-9. JTAG/Boundary Scan Pins .....	3-18
3-10. Serial EEPROM Pins .....	3-18
3-11. General Purpose I/O Pins .....	3-19
3-12. Miscellaneous Pins .....	3-21
3-13. Power and Ground Pins .....	3-24
3-14. No Connect Pins .....	3-24
3-15. Multiplexed Transparent/Non-Transparent Pins .....	3-25
4-1. PCI Clock Division Ratios .....	4-2
4-2. PLL and Clock Jitter Parameters .....	4-3
5-1. Reset Input Effect on PCI 6540 .....	5-8
5-2. Pin State during PWRGD, P_RSTIN#, S_RSTIN#, and Device Hiding .....	5-9
6-1. PCI Configuration Register Address Mapping—Transparent Mode .....	6-2
6-2. Extended Register Map—Offset from Extended Register Index, Transparent Mode .....	6-40
6-3. PCI Configuration Shadowed Registers (Used in Transparent Address Translation) .....	6-42
6-4. Extended Register Map (Used in Transparent Address Translation)— Offset from Extended Register Index .....	6-45
6-5. PCI Configuration Register Address Mapping 00h – 7Ch—Non-Transparent Mode .....	6-61
6-6. PCI Configuration Shadow Register Map—PCI Offset Used Only when CCNTRL[6]=1, Non-Transparent Mode .....	6-81
6-7. PCI Configuration Shadow Register Address Mapping 80h – FFh—Non-Transparent Mode .....	6-98
6-8. Extended Register Map—Offset from Extended Register Index, Non-Transparent Mode .....	6-118
6-9. Extended Register Map (Used in Non-Transparent Address Translation)—Offset from Extended Register Index .....	6-120
7-1. Serial EEPROM Address .....	7-3
8-1. Conventional PCI Transactions .....	8-1
8-2. Write Transaction Forwarding .....	8-2
8-3. Write Transaction Disconnect Address Boundaries .....	8-5
8-4. Read Transaction Prefetching .....	8-5
8-5. Read Prefetch Address Boundaries .....	8-6
8-6. Device Number to IDSEL S_AD Pin Mapping .....	8-10
8-7. P_SERR# Assertion Requirements in Response to Master Abort on Posted Write .....	8-13
8-8. Response to Delayed Write Target Termination .....	8-14
8-9. P_SERR# Assertion Requirements in Response to Delayed Write .....	8-14
8-10. Response to Posted Write Target Termination .....	8-15
8-11. P_SERR# Assertion Requirements in Response to Posted Write Parity Error .....	8-15
8-12. Response to Delayed Read Target Termination .....	8-16

8-13. P_SERR# Assertion Requirements in Response to Delayed Read	8-16
8-14. Response to Target Abort	8-18
9-1. Transaction Phase Definitions	9-2
9-2. Byte Lanes Assignment	9-4
9-3. PCI-X Command Encoding	9-5
9-4. Requester Attribute Bit Definitions	9-7
9-5. DEVSEL# Timing	9-9
9-6. Target Initial Latency	9-10
9-7. Split Completion Address Bit Definitions	9-12
9-8. Completer Attribute Bit Definitions	9-13
9-9. Split Completion Message Bit Definitions	9-15
9-10. PCI 6540 Error Message Indices	9-16
9-11. PCI 6540 Data Phase Signaling	9-18
9-12. Conventional PCI-to-PCI-X Command Translation	9-20
9-13. PCI-X-to-Conventional PCI Command Translation	9-21
10-1. Extended Register Map (Used in Address Translation)—Offset from Extended Register Index	10-7
10-2. PCI Configuration Shadowed Registers (Used in Address Translation)—Transparent Mode	10-8
11-1. Conventional PCI Transaction Ordering Summary	11-3
11-2. Transactions Ordering and Deadlock-Avoidance Rules	11-5
11-3. Split Transactions—Case-by-Case Discussion	11-6
12-1. Primary Interface Detected Parity Error Bit Status	12-7
12-2. Secondary Interface Detected Parity Error Bit Status	12-8
12-3. Primary Interface Data Parity Error Detected Bit Status	12-9
12-4. Secondary Interface Data Parity Error Detected Bit Status	12-10
12-5. P_PERR# Assertion	12-11
12-6. S_PERR# Assertion	12-12
12-7. P_SERR# for Data Parity Error Assertion	12-13
15-1. GPIO Pin Alternate Functions	15-2
16-1. Primary Interface Supported Commands	16-1
16-2. Secondary Interface Supported Commands	16-3
17-1. Bridge Actions for Various Cycle Types	17-1
17-2. S_IDSEL Generation	17-2
18-1. Reprogramming Prefetch Registers	18-1
19-1. Prefetched Data Timeout Flushing	19-3
21-1. States and Related Actions during Power Management Transitions	21-1
24-1. PCI 6540 JTAG IDCODE Value	24-1
24-2. JTAG Instructions (IEEE Standard 1149.1-1990)	24-1
25-1. Maximum Ratings	25-1
25-2. Functional Operating Range	25-1
25-3. DC Electrical Characteristics	25-2
25-4. PLL and Clock Jitter Parameters	25-3
25-5. 133 MHz PCI-X and 66 MHz PCI Signal Timing for Figure 25-1	25-4
26-1. PCI 6540 Mechanical Dimensions	26-2
A-1. Universal Mode Connection Example	A-6
C-1. Part Number Conversion	C-1
C-2. PCI 6520BB Versus PCI 6540BB Pin Assignment Comparison	C-1
D-1. Available Package	D-1



# REGISTERS

6-1. (PCIIDR; PCI:00h) PCI Configuration ID . . . . .	6-4
6-2. (PCICR; PCI:04h) Primary PCI Command . . . . .	6-4
6-3. (PCISR; PCI:06h) Primary PCI Status. . . . .	6-5
6-4. (PCIREV; PCI:08h) PCI Revision ID . . . . .	6-6
6-5. (PCICCR; PCI:09h – 0Bh) PCI Class Code. . . . .	6-6
6-6. (PCICLSR; PCI:0Ch) PCI Cache Line Size. . . . .	6-6
6-7. (PCILTR; PCI:0Dh) Primary PCI Bus Latency Timer. . . . .	6-6
6-8. (PCIHTR; PCI:0Eh) PCI Header Type. . . . .	6-7
6-9. (PCIBISTR; PCI:0Fh) PCI Built-In Self-Test . . . . .	6-7
6-10. (PCIPBNO; PCI:18h) PCI Primary Bus Number . . . . .	6-8
6-11. (PCISBNO; PCI:19h) PCI Secondary Bus Number. . . . .	6-8
6-12. (PCISUBNO; PCI:1Ah) PCI Subordinate Bus Number . . . . .	6-8
6-13. (PCISLTR; PCI:1Bh) Secondary PCI Bus Latency Timer . . . . .	6-8
6-14. (PCIIOBAR; PCI:1Ch) I/O Base. . . . .	6-9
6-15. (PCIIOLMT; PCI:1Dh) I/O Limit . . . . .	6-9
6-16. (PCISSR; PCI:1Eh) Secondary PCI Status. . . . .	6-10
6-17. (PCIMBAR; PCI:20h) Memory Base . . . . .	6-11
6-18. (PCIMLMT; PCI:22h) Memory Limit. . . . .	6-11
6-19. (PCIPMBAR; PCI:24h) Prefetchable Memory Base . . . . .	6-12
6-20. (PCIPMLMT; PCI:26h) Prefetchable Memory Limit. . . . .	6-12
6-21. (PCIPMBARU32; PCI:28h) Prefetchable Memory Base Upper 32 Bits. . . . .	6-13
6-22. (PCIPMLMTU32; PCI:2Ch) Prefetchable Memory Limit Upper 32 Bits. . . . .	6-13
6-23. (PCIIOBARU16; PCI:30h) I/O Base Upper 16 Bits . . . . .	6-13
6-24. (PCIIOLMTU16; PCI:32h) I/O Limit Upper 16 Bits. . . . .	6-13
6-25. (CAP_PTR; PCI:34h) New Capability Pointer . . . . .	6-14
6-26. (PCIIPR; PCI:3Dh) PCI Interrupt Pin . . . . .	6-14
6-27. (BCNTRL; PCI:3Eh) Bridge Control . . . . .	6-14
6-28. (CCNTRL; PCI:40h) Chip Control . . . . .	6-16
6-29. (DCNTRL; PCI:41h) Diagnostic Control . . . . .	6-17
6-30. (ACNTRL; PCI:42h) Arbiter Control. . . . .	6-17
6-31. (PFTCR; PCI:44h) Primary Flow-Through Control . . . . .	6-18
6-32. (TOCNTRL; PCI:45h) Timeout Control . . . . .	6-19
6-33. (MSCOPT; PCI:46h) Miscellaneous Options. . . . .	6-20
6-34. (PITLPCNT; PCI:48h) Primary Initial Prefetch Count . . . . .	6-22
6-35. (SITLPCNT; PCI:49h) Secondary Initial Prefetch Count . . . . .	6-23
6-36. (PINCPCNT; PCI:4Ah) Primary Incremental Prefetch Count . . . . .	6-23
6-37. (SINCPCNT; PCI:4Bh) Secondary Incremental Prefetch Count . . . . .	6-24
6-38. (PMAXPcnt; PCI:4Ch) Primary Maximum Prefetch Count . . . . .	6-25
6-39. (SMAXPcnt; PCI:4Dh) Secondary Maximum Prefetch Count. . . . .	6-25
6-40. (SFTCR; PCI:4Eh) Secondary Flow-Through Control. . . . .	6-26
6-41. (BUFCR; PCI:4Fh) Buffer Control . . . . .	6-27
6-42. (IACNTRL; PCI:50h) Internal Arbiter Control . . . . .	6-28
6-43. (TEST; PCI:52h) Test . . . . .	6-29
6-44. (EEPCNTRL; PCI:54h) Serial EEPROM Control. . . . .	6-29

6-45. (EEPADDR; PCI:55h) Serial EEPROM Address . . . . .	6-30
6-46. (EEPDATA; PCI:56h) Serial EEPROM Data . . . . .	6-30
6-47. (TMRCNTRL; PCI:61h) Timer Control . . . . .	6-31
6-48. (TMRcnt; PCI:62h) Timer Counter . . . . .	6-31
6-49. (PSERRRD; PCI:64h) P_SERR# Event Disable . . . . .	6-32
6-50. (GPIOOD[3:0]; PCI:65h) GPIO[3:0] Output Data . . . . .	6-33
6-51. (GPIOOE[3:0]; PCI:66h) GPIO[3:0] Output Enable . . . . .	6-33
6-52. (GPIOID[3:0]; PCI:67h) GPIO[3:0] Input Data . . . . .	6-33
6-53. (CLKCNTRL; PCI:68h) Clock Control . . . . .	6-34
6-54. (PSERRSR; PCI:6Ah) P_SERR# Status . . . . .	6-34
6-55. (CLKRUN; PCI:6Bh) Clock Run . . . . .	6-35
6-56. (PVTMBAR; PCI:6Ch) Private Memory Base . . . . .	6-36
6-57. (PVTMLMT; PCI:6Eh) Private Memory Limit . . . . .	6-36
6-58. (PVTMBARU32; PCI:70h) Private Memory Base Upper 32 Bits . . . . .	6-36
6-59. (PVTMLMTU32; PCI:74h) Private Memory Limit Upper 32 Bits . . . . .	6-36
6-60. (HSSRRRC; PCI:9Ch) Hot Swap Switch and ROR Control . . . . .	6-37
6-61. (GPIOOD[7:4]; PCI:9Dh) GPIO[7:4] Output Data . . . . .	6-38
6-62. (GPIOOE[7:4]; PCI:9Eh) GPIO[7:4] Output Enable . . . . .	6-38
6-63. (GPIOID[7:4]; PCI:9Fh) GPIO[7:4] Input Data . . . . .	6-38
6-64. (PWRUPSR; PCI:A0h) Power-Up Status . . . . .	6-39
6-65. (GPIOOD[15:8]; PCI:A1h) GPIO[15:8] Output Data . . . . .	6-39
6-66. (GPIOOE[15:8]; PCI:A2h) GPIO[15:8] Output Enable . . . . .	6-39
6-67. (GPIOID[15:8]; PCI:A3h) GPIO[15:8] Input Data . . . . .	6-39
6-68. (SCRATCHx; Ext:00h – 07h) 32-Bit Sticky Scratch . . . . .	6-41
6-69. (EXTRIDX; PCI:D3h) Extended Register Index . . . . .	6-41
6-70. (EXTRDATA; PCI:D4h) Extended Register Data . . . . .	6-41
6-71. (PCIBAR0; Primary PCI:10h) PCI Downstream I/O BAR 0 . . . . .	6-43
6-72. (PCIBAR1; Primary PCI:14h) PCI Downstream Memory BAR 1 . . . . .	6-43
6-73. (PCIBAR2; Primary PCI:18h) PCI Downstream Memory BAR 2 or Downstream Memory BAR 1 Upper 32 Bits . . . . .	6-43
6-74. (PCIUBAR0; Primary PCI:10h) PCI Upstream I/O or Memory BAR 0 . . . . .	6-44
6-75. (PCIUBAR1; Primary PCI:14h) PCI Upstream Memory BAR 1 . . . . .	6-44
6-76. (PCIUBAR2; Primary PCI:18h) PCI Upstream Memory BAR 2 or Upstream Memory Bar 1 Upper 32 Bits . . . . .	6-45
6-77. (UPSTNBAR0; Ext:08h) Upstream BAR 0 Translation Address . . . . .	6-46
6-78. (UPSTNBAR1; Ext:09h) Upstream BAR 1 Translation Address . . . . .	6-46
6-79. (UPSTNBAR2; Ext:0Ah) Upstream BAR 2 Translation Address or Upstream BAR 1 Upper 32 Bits . . . . .	6-46
6-80. (UPSBAR0MSK; Ext:0Bh) Upstream BAR 0 Translation Mask . . . . .	6-47
6-81. (UPSBAR1MSK; Ext:0Bh) Upstream BAR 1 Translation Mask . . . . .	6-47
6-82. (UPSBAR2MSK; Ext:0Bh) Upstream BAR 2 Translation Mask . . . . .	6-47
6-83. (UPSTNE; Ext:0Bh) Upstream Translation Enable . . . . .	6-48
6-84. (DWNTNBAR0; Ext:0Ch) Downstream BAR 0 Translation Address . . . . .	6-48
6-85. (DWNTNBAR1; Ext:0Dh) Downstream BAR 1 Translation Address . . . . .	6-48
6-86. (DWNTNBAR2; Ext:0Eh) Downstream BAR 2 or Downstream Memory BAR 1 Upper 32 Bits Translation Address . . . . .	6-48
6-87. (DWNBAR0MSK; Ext:0Fh) Downstream BAR 0 Translation Mask . . . . .	6-49

6-88. (DWNBAR1MSK; Ext:0Fh) Downstream BAR 1 Translation Mask . . . . .	6-49
6-89. (DWNBAR2MSK; Ext:0Fh) Downstream BAR 2 Translation Mask . . . . .	6-50
6-90. (DWNTNE; Ext:0Fh) Downstream Translation Enable . . . . .	6-50
6-91. (PMCAPID; PCI:DCh) Power Management Capability ID . . . . .	6-51
6-92. (PMNEXT; PCI:DDh) Power Management Next Capability Pointer . . . . .	6-51
6-93. (PMC; PCI:DEh) Power Management Capabilities . . . . .	6-52
6-94. (PMCSR; PCI:E0h) Power Management Control/Status . . . . .	6-53
6-95. (PMCSR_BSE; PCI:E2h) PMCSR Bridge Supports Extensions . . . . .	6-53
6-96. (PMCDATA; PCI:E3h) Power Management Data . . . . .	6-53
6-97. (HS_CNTL; PCI:E4h) Hot Swap Control . . . . .	6-54
6-98. (HS_NEXT; PCI:E5h) Hot Swap Next Capability Pointer . . . . .	6-54
6-99. (HS_CSR; PCI:E6h) Hot Swap Control/Status . . . . .	6-54
6-100. (PVPDID; PCI:E8h) Vital Product Data Capability ID . . . . .	6-55
6-101. (PVPD_NEXT; PCI:E9h) Vital Product Data Next Capability Pointer . . . . .	6-55
6-102. (PVPDAD; PCI:EAh) Vital Product Data Address . . . . .	6-55
6-103. (PVPDATA; PCI:ECh) VPD Data . . . . .	6-55
6-104. (PCIXCAPID; PCI:F0h) PCI-X Capability ID . . . . .	6-56
6-105. (PCIX_NEXT; PCI:F1h) PCI-X Next Capability Pointer . . . . .	6-56
6-106. (PCIXSSR; PCI:F2h) PCI-X Secondary Status . . . . .	6-56
6-107. (PCIXBSR; PCI:F4h) PCI-X Bridge Status . . . . .	6-58
6-108. (PCIXUPSTR; PCI:F8h) PCI-X Upstream Split Transaction . . . . .	6-59
6-109. (PCIXDNSTR; PCI:FCh) PCI-X Downstream Split Transaction . . . . .	6-59
6-110. (PCIIDR; Primary PCI:00h, Secondary PCI:40h) PCI Configuration ID . . . . .	6-62
6-111. (PCICR; Primary PCI:04h, Secondary PCI:44h) Primary PCI Command . . . . .	6-63
6-112. (PCISR; Primary PCI:06h, Secondary PCI:46h) Primary PCI Status . . . . .	6-64
6-113. (PCIREV; Primary PCI:08h, Secondary PCI:48h) PCI Revision ID . . . . .	6-65
6-114. (PCICCR; Primary PCI:09h – 0Bh, Secondary PCI:49h – 4Bh) PCI Class Code . . . . .	6-65
6-115. (PCICLSR; Primary PCI:0Ch, Secondary PCI:4Ch) Primary PCI Cache Line Size . . . . .	6-66
6-116. (PCILTR; Primary PCI:0Dh, Secondary PCI:4Dh) Primary PCI Bus Latency Timer . . . . .	6-66
6-117. (PCIHTR; Primary PCI:0Eh, Secondary PCI:4Eh) PCI Header Type . . . . .	6-66
6-118. (PCIBISTR; Primary PCI:0Fh, Secondary PCI:4Fh) PCI Built-In Self-Test . . . . .	6-66
6-119. (PCIBAR0; Primary PCI:10h, Secondary PCI:50h) PCI Downstream I/O or Memory BAR 0 . . . . .	6-67
6-120. (PCIBAR1; Primary PCI:14h, Secondary PCI:54h) PCI Downstream Memory BAR 1 . . . . .	6-68
6-121. (PCIBAR2; Primary PCI:18h, Secondary PCI:58h) PCI Downstream Memory BAR 2 or Downstream Memory BAR 1 Upper 32 Bits . . . . .	6-69
6-122. (PCISVID; Primary PCI:2Ch, Secondary PCI:6Ch) PCI Subsystem Vendor ID . . . . .	6-70
6-123. (PCISID; Primary PCI:2Eh, Secondary PCI:6Eh) PCI Subsystem ID . . . . .	6-70
6-124. (CAP_PTR; Primary PCI:34h, Secondary PCI:74h) New Capability Pointer . . . . .	6-70
6-125. (PCIPILR; PCI:3Ch, Secondary PCI:7Ch) Primary PCI Interrupt Line . . . . .	6-71
6-126. (PCIPIPR; PCI:3Dh, Secondary PCI:7Dh) Primary PCI Interrupt Pin . . . . .	6-71
6-127. (PCIPMGR; PCI:3Eh, Secondary PCI:7Eh) Primary PCI Minimum Grant . . . . .	6-71
6-128. (PCIPMLR; PCI:3Fh, Secondary PCI:7Fh) Primary PCI Maximum Latency . . . . .	6-71
6-129. (PCIIDR; Primary PCI:40h, Secondary PCI:00h) PCI Configuration ID . . . . .	6-72
6-130. (PCISCR; Primary PCI:44h, Secondary PCI:04h) Secondary PCI Command . . . . .	6-72
6-131. (PCISSR; Primary PCI:46h, Secondary PCI:06h) Secondary PCI Status . . . . .	6-73
6-132. (PCIREV; Primary PCI:48h, Secondary PCI:08h) PCI Revision ID . . . . .	6-74

6-133. (PCICCR; Primary PCI:49h – 4Bh, Secondary PCI:09h – 0Bh) PCI Class Code . . . . .	6-74
6-134. (PCISCLSR; Primary PCI:4Ch, Secondary PCI:0Ch) Secondary PCI Cache Line Size . . . . .	6-75
6-135. (PCISLTR; Primary PCI:4Dh, Secondary PCI:0Dh) Secondary PCI Bus Latency Timer . . . . .	6-75
6-136. (PCIHTR; Primary PCI:4Eh, Secondary PCI:0Eh) PCI Header Type . . . . .	6-75
6-137. (PCIBISTR; Primary PCI:4Fh, Secondary PCI:0Fh) PCI Built-In Self-Test . . . . .	6-75
6-138. (PCIUBAR0; Primary PCI:50h, Secondary PCI:10h) PCI Upstream I/O or Memory BAR 0 . . . . .	6-76
6-139. (PCIUBAR1; Primary PCI:54h, Secondary PCI:14h) PCI Upstream Memory BAR 1 . . . . .	6-77
6-140. (PCIUBAR2; Primary PCI:58h, Secondary PCI:18h) PCI Upstream Memory BAR 2 or Upstream Memory Bar 1 Upper 32 Bits . . . . .	6-78
6-141. (PCISVID; Primary PCI:6Ch, Secondary PCI:2Ch) PCI Subsystem Vendor ID . . . . .	6-79
6-142. (PCISID; Primary PCI:6Eh, Secondary PCI:2Eh) PCI Subsystem ID . . . . .	6-79
6-143. (CAP_PTR; Primary PCI:74h, Secondary PCI:34h) New Capability Pointer . . . . .	6-79
6-144. (PCISILR; PCI:7Ch, Secondary PCI:3Ch) Secondary PCI Interrupt Line . . . . .	6-80
6-145. (PCISIPR; PCI:7Dh, Secondary PCI:3Dh) Secondary PCI Interrupt Pin . . . . .	6-80
6-146. (PCISMGR; PCI:7Eh, Secondary PCI:3Eh) Secondary PCI Minimum Grant . . . . .	6-80
6-147. (PCISMLR; PCI:7Fh, Secondary PCI:3Fh) Secondary PCI Maximum Latency . . . . .	6-80
6-148. (BCNTRL; PCI:42h) Bridge Control . . . . .	6-82
6-149. (PFTCR; PCI:44h) Primary Flow-Through Control . . . . .	6-84
6-150. (TOCNTRL; PCI:45h) Timeout Control . . . . .	6-85
6-151. (MSCOPT; PCI:46h) Miscellaneous Options . . . . .	6-86
6-152. (PITLPCNT; PCI:48h) Primary Initial Prefetch Count . . . . .	6-88
6-153. (SITLPCNT; PCI:49h) Secondary Initial Prefetch Count . . . . .	6-89
6-154. (PINPCNT; PCI:4Ah) Primary Incremental Prefetch Count . . . . .	6-89
6-155. (SINPCNT; PCI:4Bh) Secondary Incremental Prefetch Count . . . . .	6-90
6-156. (PMAXPCNT; PCI:4Ch) Primary Maximum Prefetch Count . . . . .	6-91
6-157. (SMAXPCNT; PCI:4Dh) Secondary Maximum Prefetch Count . . . . .	6-91
6-158. (SFTCR; PCI:4Eh) Secondary Flow-Through Control . . . . .	6-92
6-159. (BUFCR; PCI:4Fh) Buffer Control . . . . .	6-93
6-160. (IACNTRL; PCI:50h) Internal Arbiter Control . . . . .	6-94
6-161. (TEST; PCI:52h) Test . . . . .	6-95
6-162. (EEPCNTRL; PCI:54h) Serial EEPROM Control . . . . .	6-95
6-163. (EEPADDR; PCI:55h) Serial EEPROM Address . . . . .	6-96
6-164. (EEPDATA; PCI:56h) Serial EEPROM Data . . . . .	6-96
6-165. (TMRCNTRL; PCI:61h) Timer Control . . . . .	6-97
6-166. (TMRcnt; PCI:62h) Timer Counter . . . . .	6-97
6-167. (XBDWNCA; PCI:80h) Cross-Bridge Downstream Configuration Address . . . . .	6-100
6-168. (XBDWNCD; PCI:84h) Cross-Bridge Downstream Configuration Data . . . . .	6-100
6-169. (XBUPSCA; PCI:88h) Cross-Bridge Upstream Configuration Address . . . . .	6-100
6-170. (XBUPSCD; PCI:8Ch) Cross-Bridge Upstream Configuration Data . . . . .	6-100
6-171. (XBDWNCOS; PCI:90h) Cross-Bridge Downstream Configuration Ownership Semaphore . . . . .	6-101
6-172. (XBUPSCOS; PCI:91h) Cross-Bridge Upstream Configuration Ownership Semaphore . . . . .	6-101
6-173. (XBCOS; PCI:92h) Cross-Bridge Configuration Ownership Status . . . . .	6-101
6-174. (CLKCNTRL; PCI:94h) Clock Control . . . . .	6-102
6-175. (PSERRED; PCI:96h) P_SERR# Event Disable . . . . .	6-103
6-176. (SSERRED; PCI:97h) S_SERR# Event Disable . . . . .	6-104
6-177. (PSSERRSR; PCI:98h) P_SERR# and S_SERR# Status . . . . .	6-105

6-178. (GPIOOD[3:0]; PCI:99h) GPIO[3:0] Output Data . . . . .	6-106
6-179. (GPIOOE[3:0]; PCI:9Ah) GPIO[3:0] Output Enable. . . . .	6-106
6-180. (GPIOID[3:0]; PCI:9Bh) GPIO[3:0] Input Data. . . . .	6-106
6-181. (HSSRR; PCI:9Ch) Hot Swap Switch and ROR Control. . . . .	6-107
6-182. (GPIOOD[7:4]; PCI:9Dh) GPIO[7:4] Output Data . . . . .	6-108
6-183. (GPIOOE[7:4]; PCI:9Eh) GPIO[7:4] Output Enable. . . . .	6-108
6-184. (GPIOID[7:4]; PCI:9Fh) GPIO[7:4] Input Data. . . . .	6-108
6-185. (PWRUPSR; PCI:A0h) Power-Up Status . . . . .	6-109
6-186. (GPIOOD[15:8]; PCI:A1h) GPIO[15:8] Output Data . . . . .	6-109
6-187. (GPIOOE[15:8]; PCI:A2h) GPIO[15:8] Output Enable. . . . .	6-109
6-188. (GPIOID[15:8]; PCI:A3h) GPIO[15:8] Input Data. . . . .	6-109
6-189. (UPSMSG0; PCI:A4h) Upstream Message 0 . . . . .	6-110
6-190. (UPSMSG1; PCI:A5h) Upstream Message 1 . . . . .	6-110
6-191. (UPSMSG2; PCI:A6h) Upstream Message 2 . . . . .	6-110
6-192. (UPSMSG3; PCI:A7h) Upstream Message 3 . . . . .	6-110
6-193. (DWNMSG0; PCI:A8h) Downstream Message 0 . . . . .	6-111
6-194. (DWNMSG1; PCI:A9h) Downstream Message 1 . . . . .	6-111
6-195. (DWNMSG2; PCI:AAh) Downstream Message 2 . . . . .	6-111
6-196. (DWNMSG3; PCI:ABh) Downstream Message 3 . . . . .	6-111
6-197. (MSICAPID; PCI:ACH) Message Signaled Interrupts Capability ID. . . . .	6-112
6-198. (MSINEXT; PCI:ADh) Message Signaled Interrupts Next Capability Pointer . . . . .	6-112
6-199. (MSIC; PCI:AEh) Message Signaled Interrupts Control . . . . .	6-112
6-200. (MSIADDR; PCI:B0h) Message Signaled Interrupts Address . . . . .	6-113
6-201. (MSIUADDR; PCI:B4h) Message Signaled Interrupts Upper Address . . . . .	6-113
6-202. (MSIDATA; PCI:B8h) Message Signaled Interrupts Data . . . . .	6-113
6-203. (DWNDBIE; PCI:C0h) Downstream Doorbell Interrupt Enable . . . . .	6-114
6-204. (DWNDBIR; PCI:C2h) Downstream Doorbell Interrupt Request . . . . .	6-114
6-205. (UPSDBIE; PCI:C4h) Upstream Doorbell Interrupt Enable . . . . .	6-114
6-206. (UPSDBIR; PCI:C6h) Upstream Doorbell Interrupt Request . . . . .	6-114
6-207. (DWNDBIS; PCI:C8h) Downstream Doorbell Interrupt Status. . . . .	6-115
6-208. (DWNINTSR; PCI:CAh) Downstream Interrupt Status . . . . .	6-115
6-209. (UPSINTE; PCI:CBh) Upstream Interrupt Enable . . . . .	6-115
6-210. (UPSDBIS; PCI:CCh) Upstream Doorbell Interrupt Status . . . . .	6-116
6-211. (UPSINTSR; PCI:CEh) Upstream Interrupt Status . . . . .	6-116
6-212. (DWNINTE; PCI:CFh) Downstream Interrupt Enable . . . . .	6-116
6-213. (NTCOS; PCI:D2h) Non-Transparent Configuration Ownership Semaphore . . . . .	6-117
6-214. (SCRATCHx; Ext:00h – 07h) 32-Bit Sticky Scratch. . . . .	6-119
6-215. (EXTRIDX; PCI:D3h) Extended Register Index. . . . .	6-119
6-216. (EXTRDATA; PCI:D4h) Extended Register Data . . . . .	6-119
6-217. (UPSTNBAR0; Ext:08h) Upstream BAR 0 Translation Address . . . . .	6-121
6-218. (UPSTNBAR1; Ext:09h) Upstream BAR 1 Translation Address . . . . .	6-121
6-219. (UPSTNBAR2; Ext:0Ah) Upstream BAR 2 Translation Address or Upstream BAR 1 Upper 32 Bits . . . . .	6-121
6-220. (UPSBAR0MSK; Ext:0Bh) Upstream BAR 0 Translation Mask. . . . .	6-122
6-221. (UPSBAR1MSK; Ext:0Bh) Upstream BAR 1 Translation Mask. . . . .	6-122
6-222. (UPSBAR2MSK; Ext:0Bh) Upstream BAR 2 Translation Mask. . . . .	6-122

6-223. (UPSTNE; Ext:0Bh) Upstream Translation Enable . . . . .	6-123
6-224. (DWNTNBAR0; Ext:0Ch) Downstream BAR 0 Translation Address . . . . .	6-124
6-225. (DWNTNBAR1; Ext:0Dh) Downstream BAR 1 Translation Address . . . . .	6-124
6-226. (DWNTNBAR2; Ext:0Eh) Downstream BAR 2 or Downstream Memory BAR 1 Upper 32 Bits Translation Address . . . . .	6-124
6-227. (DWNBAR0MSK; Ext:0Fh) Downstream BAR 0 Translation Mask . . . . .	6-125
6-228. (DWNBAR1MSK; Ext:0Fh) Downstream BAR 1 Translation Mask . . . . .	6-125
6-229. (DWNBAR2MSK; Ext:0Fh) Downstream BAR 2 Translation Mask . . . . .	6-125
6-230. (DWNTNE; Ext:0Fh) Downstream Translation Enable . . . . .	6-126
6-231. (CCNTRL; PCI:D8h) Chip Control . . . . .	6-127
6-232. (DCNTRL; PCI:D9h) Diagnostic Control . . . . .	6-128
6-233. (ACNTRL; PCI:DAh) Arbiter Control . . . . .	6-128
6-234. (PMCAPID; PCI:DCh) Power Management Capability ID . . . . .	6-129
6-235. (PMNEXT; PCI:DDh) Power Management Next Capability Pointer . . . . .	6-129
6-236. (PMC; PCI:DEh) Power Management Capabilities . . . . .	6-130
6-237. (PMCSR; PCI:E0h) Power Management Control/Status . . . . .	6-131
6-238. (PMCSR_BSE; PCI:E2h) PMCSR Bridge Supports Extensions . . . . .	6-131
6-239. (PMCDATA; PCI:E3h) Power Management Data . . . . .	6-131
6-240. (HS_CNTL; PCI:E4h) Hot Swap Control . . . . .	6-132
6-241. (HS_NEXT; PCI:E5h) Hot Swap Next Capability Pointer . . . . .	6-132
6-242. (HS_CSR; PCI:E6h) Hot Swap Control/Status . . . . .	6-132
6-243. (PVPDID; PCI:E8h) Vital Product Data Capability ID . . . . .	6-133
6-244. (PVPD_NEXT; PCI:E9h) Vital Product Data Next Capability Pointer . . . . .	6-133
6-245. (PVPDAD; PCI:EAh) Vital Product Data Address . . . . .	6-133
6-246. (PVPDATA; PCI:ECh) VPD Data. . . . .	6-133
6-247. (PCIXCAPID; PCI:F0h) PCI-X Capability ID . . . . .	6-134
6-248. (PCIX_NEXT; PCI:F1h) PCI-X Next Capability Pointer . . . . .	6-134
6-249. (PCIX_CMD; PCI:F2h) PCI-X Command . . . . .	6-134
6-250. (PCIXBSR; PCI:F4h) PCI-X Bridge Status . . . . .	6-135
6-251. (PCIXUPSTR; PCI:F8h) PCI-X Upstream Split Transaction . . . . .	6-136
6-252. (PCIXDNSTR; PCI:FCh) PCI-X Downstream Split Transaction . . . . .	6-136

# PREFACE

The information contained in this document is subject to change without notice. Although an effort has been made maintain accurate information, there may be misleading or even incorrect statements made herein.

## Supplemental Documentation

The following is a list of documentation to provide further details:

- *PCI 6520BB and PCI 6540BB Pin Comparison, Application Note, Version 1.1*, August 5, 2003  
PLX Technology, Inc.  
870 Maude Avenue  
Sunnyvale, California 94085 USA  
Tel: 408 774-9060 or 800 759-3735, Fax: 408 774-2169, <http://www.plxtech.com/>
- *PCI Local Bus Specification, Revision 2.1*, June 1, 1995  
PCI Special Interest Group (PCI-SIG)  
5440 SW Westgate Drive #217, Portland, OR 97221 USA  
Tel: 503 291-2569, Fax: 503 297-1090, <http://www.pcisig.com>
- *PCI Local Bus Specification, Revision 2.3*  
PCI Special Interest Group (PCI-SIG)  
5440 SW Westgate Drive #217, Portland, OR 97221 USA  
Tel: 503 291-2569, Fax: 503 297-1090, <http://www.pcisig.com>
- *PCI to PCI Bridge Architecture Specification, Revision 1.1*  
PCI Special Interest Group (PCI-SIG)  
5440 SW Westgate Drive #217, Portland, OR 97221 USA  
Tel: 503 291-2569, Fax: 503 297-1090, <http://www.pcisig.com>
- *PCI Bus Power Management Interface Specification, Revision 1.1*, December 18, 1998  
PCI Special Interest Group (PCI-SIG)  
5440 SW Westgate Drive #217, Portland, OR 97221 USA  
Tel: 503 291-2569, Fax: 503 297-1090, <http://www.pcisig.com>
- *PICMG 2.1, R2.0, CompactPCI Hot Swap Specification*, January 2001  
PCI Industrial Computer Manufacturers Group (PICMG)  
c/o Virtual Inc., 401 Edgewater Place, Suite 500, Wakefield, MA 01880, USA  
Tel: 781 246-9318, Fax: 781 224-1239, <http://www.picmg.org>
- *PCI-X Addendum to PCI Local Bus Specification, Revision 1.0b*  
PCI Special Interest Group (PCI-SIG)  
5440 SW Westgate Drive #217, Portland, OR 97221 USA  
Tel: 503 291-2569, Fax: 503 297-1090, <http://www.pcisig.com>
- IEEE Standard 1149.1-1990, *IEEE Standard Test Access Port and Boundary-Scan Architecture*, 1990  
The Institute of Electrical and Electronics Engineers, Inc.  
445 Hoes Lane, PO Box 1331, Piscataway, NJ 08855-1331, USA  
Tel: 800 678-4333 (domestic only) or 732 981-0060, Fax: 732 981-1721, <http://www.ieee.org>

## Preface

---

**Note:** In this data book, shortened titles are provided to the previously listed documents. The following table lists these abbreviations.

### Supplemental Documentation Abbreviations

Abbreviation	Document
<i>PCI r2.1</i>	<i>PCI Local Bus Specification, Revision 2.1</i>
<i>PCI r2.3</i>	<i>PCI Local Bus Specification, Revision 2.3</i>
<i>P-to-P Bridge r1.1</i>	<i>PCI to PCI Bridge Architecture Specification, Revision 1.1</i>
<i>PCI Power Mgmt. r1.1</i>	<i>PCI Bus Power Management Interface Specification, Revision 1.1</i>
<i>PICMG 2.1 R2.0</i>	<i>PICMG 2.1 R2.0 CompactPCI Hot Swap Specification</i>
<i>PCI-X r1.0b</i>	<i>PCI-X Addendum to PCI Local Bus Specification, Revision 1.0b</i>
IEEE Standard 1149.1-1990	<i>IEEE Standard Test Access Port and Boundary-Scan Architecture</i>

## DATA ASSIGNMENT CONVENTIONS

### Data Assignment Conventions

Data Width	PCI 6540 Convention
1 byte (8 bits)	Byte
2 bytes (16 bits)	Word
4 bytes (32 bits)	DWORD/Dword
8 bytes (64 bits)	QWORD/Qword

## REVISION HISTORY

Date	Version	Comments
11/03	1.0	Production Release, Silicon Revision BB.



## FEATURE SUMMARY

The PLX PCI 6540 PCI-X-to-PCI-X Bridge is a universal device capable of 64-bit, 133 MHz operation. The device is designed for high-performance and high-availability uses such as PCI-X slot expansion, PCI-X-to-PCI conversion, multi-device attachment, frequency conversion, high-availability Hot Swap, and universal system-to-system bridging. The PCI 6540 has sophisticated buffer management and buffer configuration options designed to provide customizable performance for efficient throughput, and is the *only* PCI-X-to-PCI-X bridge that offers *true non-transparent* operation.

- PCI-X Rev 1.0-compliant at 64-bits, 133 MHz
  - Backward compatible with *PCI r2.3*
  - Support for input-pin-enforced PCI-X operation
- 5V tolerant I/O
- Asynchronous design for primary and secondary ports
  - 33 to 133 MHz operation
  - Either port may run at the higher frequency
- 16 GPIO pins with output control
  - Eight pins have power-up status latch capabilities
- *PICMG 2.1 R2.0* with  $PI = 1$ 
  - Support for device hiding
- Primary port can be set to PCI-X protocol, without requiring the normal reset initialization sequence
- Flow-through, zero wait state bursts of up to 4 KB
  - Optimal for large volume Data transfer
- Supports up to four simultaneous Posted Writes and Delayed transactions in each direction
- Supports up to four simultaneous Split transactions in each direction in PCI-X mode
- Optional segmented 1-KB buffer for each of the four Read FIFO entries
- 10-KB buffers
  - 1-KB downstream Posted Write buffer
  - 1-KB upstream Posted Write buffer
  - 4-KB downstream Read Data buffer
  - 4-KB upstream Read Data buffer
- Configurable prefetch size of up to 2 KB
  - Ideal for PCI-to-PCI-X transfers
- Buffer management allowing timed flush of FIFO
- 5 secondary clock outputs
  - Pin controlled enable
  - Individual maskable control
- Supports downstream and upstream Lock
- Supports secondary port PCI/PCI-X Private Devices and Private Memory space
- Reference clock input option
  - Primary and secondary port PCI-X frequency detection
- Serial EEPROM loadable
- Programmable *PCI Read-Only* register configurations
- Programmable arbitration for eight bus masters on the secondary interface
  - Optional External Arbiter
- PCI Mobile Design Guide and Power Management  $D_{3cold}$  Wakeup capable
  - PME# support
- Enhanced address decoding
  - Supports 32-bit I/O address range
  - 32-bit Memory-Mapped I/O Address range
  - ISA-Aware mode for legacy support in the first 64 KB of I/O address range
  - VGA addressing and palette snooping support

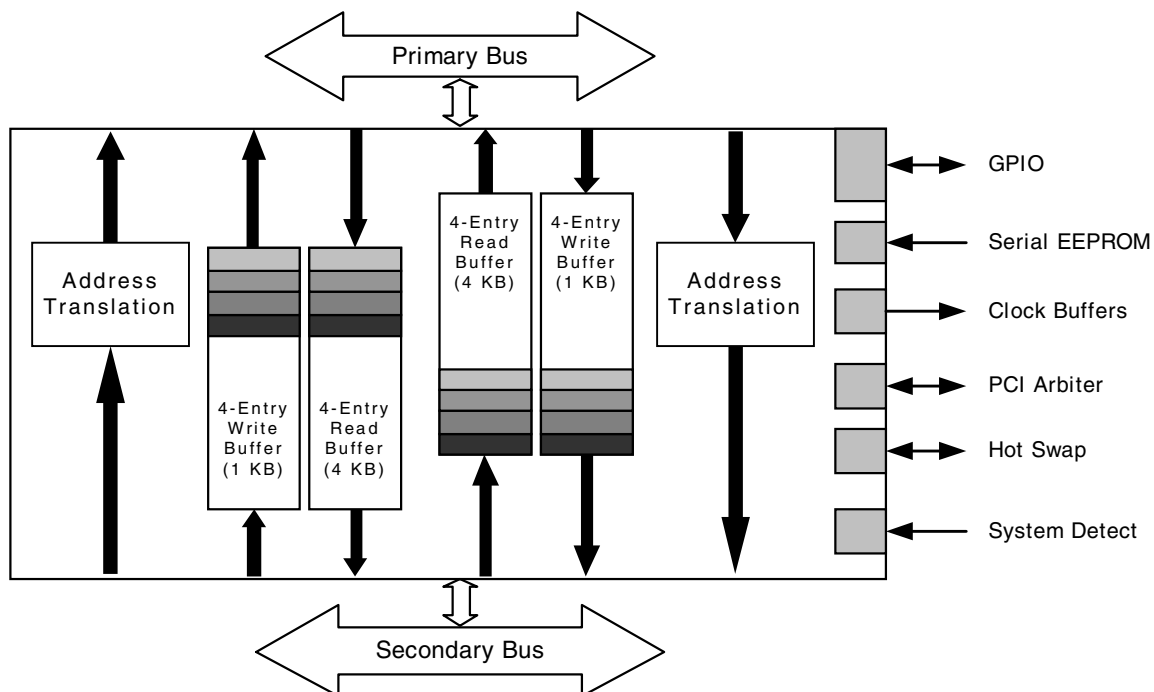
## Feature Summary

---

- IEEE Standard 1149.1-1990 JTAG interface
- Low power 0.25 $\mu$  CMOS process
- Industry standard 27 x 27 mm 380-pin (ball) PBGA package

### ***PCI 6540 Non-Transparent and Universal Mode Features***

- Programmable *Transparent*, *Non-Transparent*, or *Universal* mode operation
  - Jumperless switching between system slot and peripheral slot applications in a CompactPCI system (Universal mode)
  - Address Remapping to secondary PCI-X Bus
  - Pin-selectable primary or secondary port system boot-up priority
  - Optional default 16 MB Memory space capability, to avoid Initially Retry or Initially Not Respond requirement
  - Semaphore-backed Cross-Bridge Configuration space access
  - Powerful multi-source (direct encoded, doorbell, PCI-X reset, external pin) programmable interrupt pins for primary and secondary ports
  - Behaves as a Memory-Mapped PCI-X device
  - Primary and secondary port controllable GPIOs
  - Available primary and secondary Power Status inputs for port power detection
  - *Power Good* input for full chip reset
  - Independent primary and secondary port Reset inputs
  - Configurable primary and secondary Reset outputs
  - Sticky Scratch registers, immune to PCI-X resets
- 



PCI 6540 Block Diagram

# 1 INTRODUCTION

This section provides information about PLX Technology, Inc., and its products, the PCI 6000 Bridge Series, and PCI 6540 features and applications.

## 1.1 COMPANY AND PRODUCT INFORMATION

PLX Technology, Inc., is the leading supplier of standard interconnect silicon to the storage, communications, server, and embedded-control industries. PLX's comprehensive I/O interconnect product offering ranges from I/O accelerators, PCI-to-PCI bridges, PCI-X-to-PCI-X bridges, and HyperTransport™ bridges to the PLX PCI Express-based family of switches and bridges currently under development.

In addition to a broad product offering, PLX provides development tool support through Software Development Kits (SDKs), hardware Rapid Development Kits (RDKs), and third-party tool support through the PLX Partner Program. Our complete tool offering, combined with leadership PLX silicon, enables system designers to maximize system throughput, lower development costs, minimize system design risk, and provide faster time to market.

The PLX commitment to meeting customer requirements extends beyond complete product solutions, and includes active participation in industry associations. PLX contributes to the key standard-setting bodies in our industry, including PCI-SIG™ (the special interest group responsible for the creation and release of all PCI specifications), PICMG® (the organization responsible for CompactPCI and the new AdvancedTCA™ standard for fabrics), HyperTransport™ Consortium, and Blade Systems Alliance (BladeS). Furthermore, PLX is a key developer for PCI Express technology and a member of the Intel Developers Network for PCI Express Technology.

Founded in 1986, PLX has been developing products based on the PCI industry standard since 1994. PLX is publicly traded (NASDAQ:PLXT) and headquartered in Sunnyvale, CA, USA, with other domestic offices in Utah and Southern California. PLX European operations are based in the United Kingdom and Asian operations are based in China and Japan.

## 1.2 PCI 6000 BRIDGE SERIES

The PLX PCI 6000 series offers the industry's broadest set of PCI-to-PCI and PCI-X-to-PCI-X bridges. These bridges allow additional devices to be attached to the PCI Bus, and provide the ability to include intelligent adapters on a PCI Bus. In addition, these bridges allow PCI Buses of different speeds to be part of the same subsystem.

The PLX PCI and PCI-X family of interconnect products include both PCI-to-PCI and PCI-X-to-PCI-X bridging devices, offering system designers innovative features along with improved I/O performance. The PLX PCI 6000 series of PCI-to-PCI bridging products provide support for the entire range of current PCI Bus data widths and speeds, including 32-bit 33 MHz, 64-bit 66 MHz, and the latest 64-bit 133 MHz PCI-X variety of the standard.

The PCI 6000 product line is distinguished by featuring the widest range of options, lowest power, highest performance, and smallest footprints in the industry. The product line includes features such as the ability to clock the PCI Bus segments asynchronously to one another, and operate the component in Transparent or *true* Non-Transparent mode. Universal mode is crucial when the same module is designed to be used as a system host or peripheral.

The entire line of PLX bridging products are designed to provide high-performance interconnect for servers, storage, telecommunications, networking, and embedded applications. Like all PLX interconnect chips, the PCI 6000 series products are supported by PLX comprehensive reference design tools and the industry-recognized PLX support infrastructure.

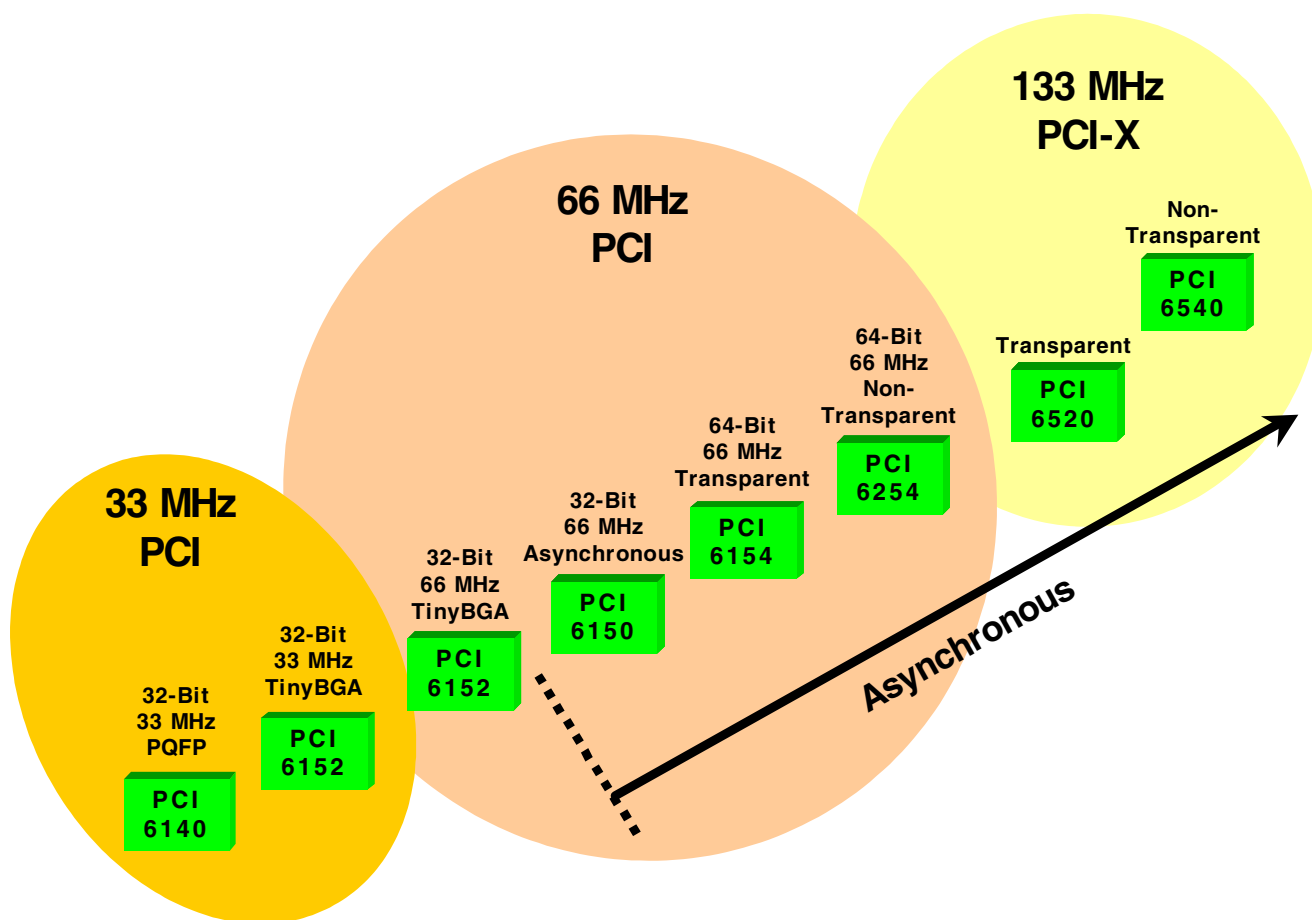


Figure 1-1. PCI 6000 Series of PCI-to-PCI Bridges

## 1.2.1 PCI 6540

The PCI 6540 is the most powerful *PCI-X-to-PCI-X* bridging device offered in the industry. As illustrated in Figure 1-2, the PCI 6540 is a two-port device providing full *asynchronous* operation between the *primary* and *secondary* ports. The secondary bus may be run faster than the primary bus, and vice versa.

The PCI 6540 is a dual-mode device, providing both *Transparent* and *Non-Transparent* operation in a single product, and can be used in *Universal* mode, providing the ability to operate the product in a CompactPCI system slot or peripheral slot.

*PCI-X* is the next generation of the industry-standard PCI Bus. This standard provides complete backward compatibility in terms of electrical characteristics, software, and form factor, and when used in *PCI-X* mode, the maximum transfer speed increases to 133 MHz, allowing a maximum bursting bandwidth of 1 Gbps. Alternatively, *PCI-X* allows greater electrical expansion of the PCI Bus at 66 MHz by registering Data transfers.

A *Transparent* PCI bridge is meant to provide electrical isolation to the system. It allows additional loads (and devices) to be attached to the bus, and can also be used to operate dissimilar PCI Bus data widths and speeds on the same system. *For example*, a *Transparent* bridge can allow several 64-bit, 66 MHz PCI devices to attach to a 133 MHz *PCI-X* slot.

A *Non-Transparent* PCI bridge offers address isolation in addition to electrical isolation. Devices on both sides of the bridge retain their own independent Memory space, and data from one side of the bridge is forwarded to the other side, using an address translation mechanism. A *Non-Transparent* bridge is used when there is more than one intelligent entity (*such as* multiple processors) in the system. It is a common mechanism used for creating intelligent I/O cards and multi-processor systems.

An *asynchronous* bridge provides the ability to run each port from a completely independent clock. This allows the system designer to provide the highest performance on each side of the bridge, without forcing one side to slow down based on a slower device on the other side of the bridge. The advantage of an asynchronous bridge is that the two clock domains can be arbitrarily different, and not based on a synchronous version of the other clock.

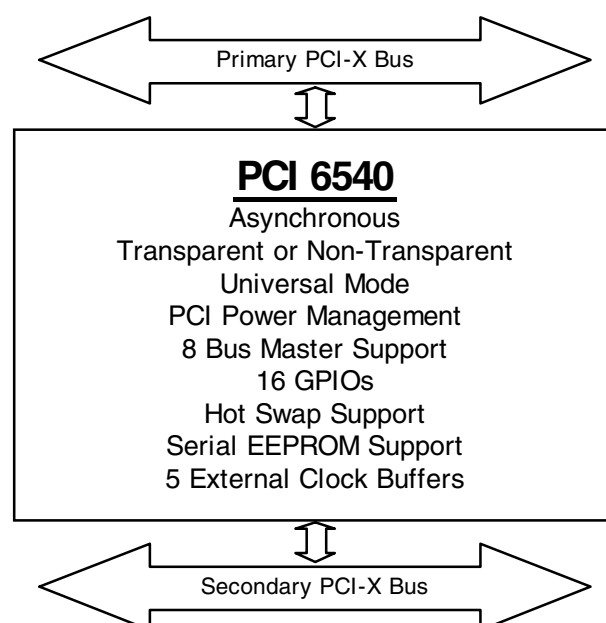


Figure 1-2. PCI 6540 PCI-X-to-PCI-X Bridge

## 1.3 FEATURE DESCRIPTION

The PCI 6540 provides a range of added-value features to system designers, including:

- Two PCI-X ports, each capable of running at the full 64-bit, 133 MHz speed
  - Backward compatible to *PCI r2.3*
- Asynchronous primary and secondary ports
  - Ports can operate from 33 MHz to 133 MHz
  - Either port can operate slower or faster than the other port
- Transparent or Non-Transparent mode operation
- Universal mode operation
  - Allows the same card to operate both as a system or peripheral card in a CompactPCI chassis
- 5V tolerant I/O
- Programmable prefetch
- Programmable Flow Through
- Zero wait state burst
- 10 KB Data FIFO
- Five secondary clock outputs
- Reference clock input for frequency detection
- PCI Power Management support
- Arbitration support for eight secondary bus masters
- Serial EEPROM for configuration
- 16 General Purpose I/O pins
- Hot Swap *Ready*
- Vital Product Data (VPD)
- JTAG boundary scan

## 1.4 APPLICATIONS

### 1.4.1 Multiple Device Expansion

Figure 1-3 illustrates the PCI 6540 used to provide electrical isolation to the PCI-X Bus. This is necessary because PCI-X slots have a restriction on the number of loads that can be accommodated. The devices on the secondary port can be *PCI* or *PCI-X*, and the bus can operate at a range of frequencies and bus data widths. This configuration is a common mechanism for providing multiple, high-speed Ethernet or Fibre Channel connections on a single PCI-X card.

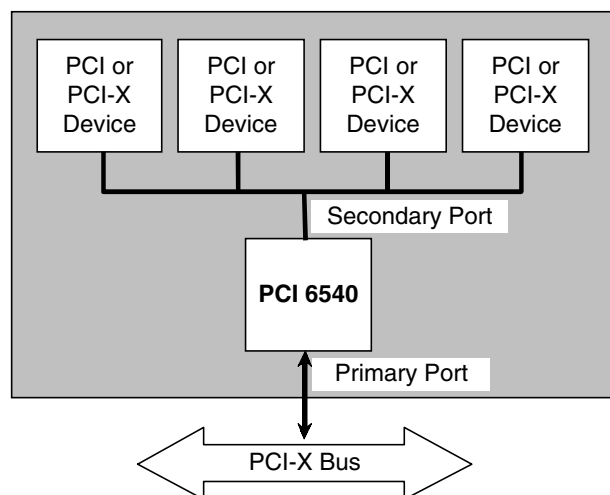


Figure 1-3. Multiple Device Expansion

## 1.4.2 Intelligent Adapters

Figure 1-4 illustrates how the PCI 6540 can be used to allow multiple CPUs to be included in a single system. Because the host (not shown) and card CPU both expect to enumerate and control the entire address space, the PCI 6540 bridge must isolate the address spaces. This configuration is used to create RAID controller cards, and intelligent adapter cards of various types in storage and communication subsystems.

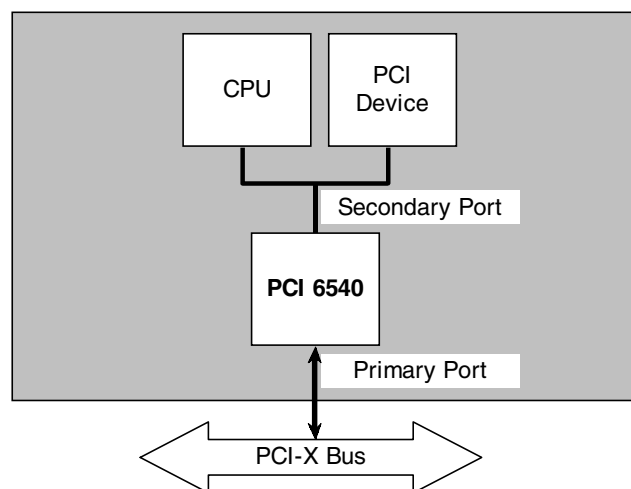


Figure 1-4. Intelligent Adapters

## 1.4.3 CompactPCI Universal Application

Figure 1-5 illustrates the PCI 6540 being used in *Universal* mode. In this application, the same card can be used for a system or peripheral slot in a CompactPCI backplane without jumpers. The PCI 6540 senses the type of slot (system or peripheral) and configures itself as *Transparent* or *Non-Transparent*, as appropriate.

In the system slot, the CPU is expected to operate as a host, and the PCI 6540 operates in Transparent mode. In the peripheral slot, the CPU is part of an intelligent subsystem, and the PCI 6540 is configured in Non-Transparent mode.

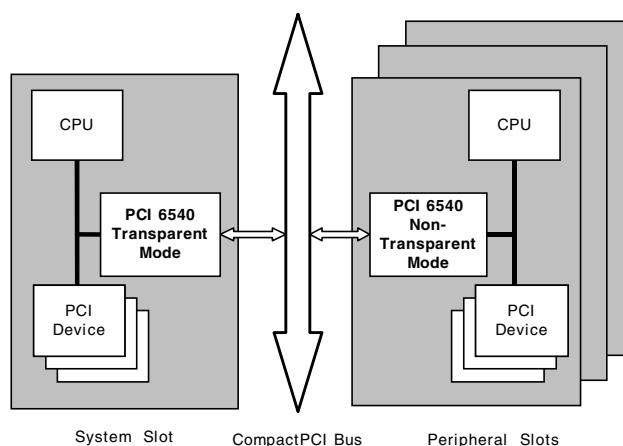


Figure 1-5. CompactPCI Universal Application





## 2 FUNCTIONAL OVERVIEW

This section describes general operation of the PCI 6540 bridge, and provides an overview of write and read transactions, and Non-Transparent mode.

### 2.1 GENERAL OPERATION

As illustrated in Figure 2-1, the PCI 6540 uses programmable buffers to regulate data flow between the primary and secondary ports. There are two sets of buffers—one for *downstream* commands (data flows from primary-to-secondary bus) and one for *upstream* commands (data flows from secondary-to-primary bus). The buffers are organized as follows:

- 4-Entry Read buffer (4 KB)
- 4-Entry Write buffer (1 KB)

Each PCI and/or PCI-X port can run at different (asynchronous) frequencies, which allows the designer to optimize the performance of each bus. Both the primary and secondary PCI and/or PCI-X ports may be operated at 32- or 64-bit bus data widths, and the two buses may be of different widths.

The PCI 6540 provides an Internal Arbiter function on the secondary bus, for up to eight secondary bus masters. However, the Internal Arbiter may be disabled if an External Arbiter is used. The PCI 6540 also sources five secondary PCI and/or PCI-X clock outputs.

The PCI 6540 is CompactPCI Hot Swap *Ready*, and complies with *PICMG 2.1 R2.0* with *High Availability* Programming Interface level 1 (PI = 1). (Refer to Section 22, “Hot Swap,” for further details.)

The PCI 6540 provides features satisfying the requirements of *PCI Power Mgmt. r1.1*, supporting Power Management states  $D_0$  through  $D_{3cold}$  and  $D_{3hot}$ . Both the primary and secondary PCI and/or PCI-X ports provide a PME# pin. (Refer to Section 21, “Power Management,” for further details.)

The PCI 6540 supports a serial EEPROM device for register configuration data. This allows the PCI 6540 to automatically load custom configuration upon power-up, which minimizes the software overhead of configuring the bridge through a host processor.

The PCI 6540 fully supports Vital Product Data (VPD) by providing the Address, Data, and Control registers (PVPDAD; PCI:EAh, PVPDATA; PCI:ECb, PVPDID; PCI:E8h, and PVPD\_NEXT; PCI:E9h) for accessing VPD stored in the unused portion of the serial EEPROM. VPD allows reading or writing of user data to the upper 192 bytes of serial EEPROM space, and that data can contain information such as board serial number, software revision, firmware revision, or other data required for non-volatile storage. (Refer to Section 23, “VPD,” for further details.)

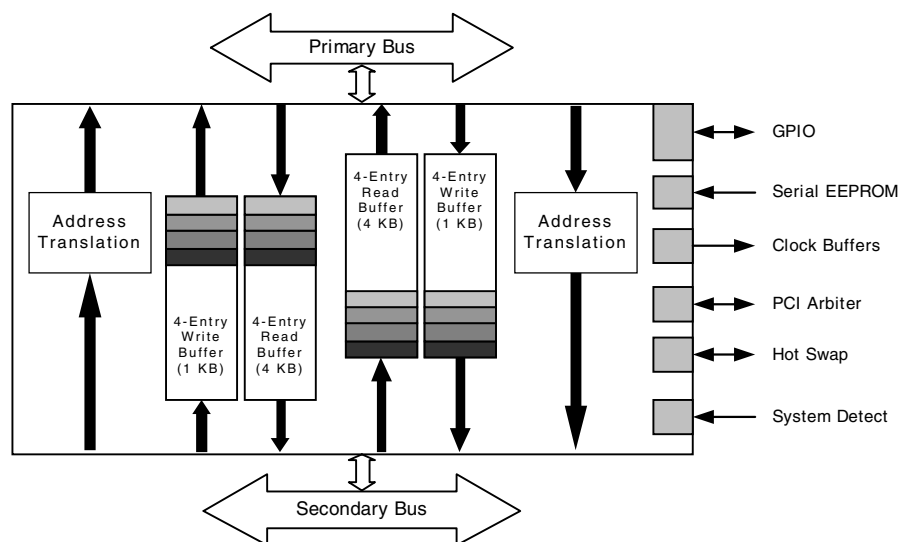


Figure 2-1. PCI 6540 Block Diagram

## 2.2 WRITE TRANSACTIONS

The primary or secondary bus accomplishes a Write operation by placing the address and data into the *Write buffer*. This initiates a PCI Write operation on the other bus. The Write operation is called a *Posted Write* operation, because the initiating bus performs the write, then moves on without waiting for the operation to complete.

The PCI 6540 provides the ability to combine Write operations when the operations are directed at the same Address range. The device recognizes when Write operations are directed at consecutive addresses, and accumulates and bursts those Write operations to the PCI Bus for increased bandwidth.

In addition, the PCI 6540 has the capability to start a Write operation before receiving all Write data. In this case, the Write operation begins when there is sufficient Write data to begin the burst, providing a Flow-Through operation as the balance of the Write data arrives in the device.

## 2.3 READ TRANSACTIONS

When the downstream or upstream bus needs to read data from the other bus, the bus places the Read request into the *Read Command Queue*. This initiates a Read operation on the other bus, and the data is placed into the associated *Read buffer* as it returns. For PCI-X transactions, the PCI 6540 also prefetches data as defined in the original Read operation.

For PCI transactions, there is an additional prefetch mechanism when returning the requested Read data. In this mode, the PCI 6540 can be programmed to prefetch up to 2 KB of data at a time. This data is stored in the Read buffer and is not flushed until the buffer times out. If requested, prefetched data can be

delivered to the PCI Bus without the normal read on the other bus.

## 2.4 NON-TRANSPARENT MODE

The PCI 6540 controls Non-Transparent Read and Write operations in a similar way to Transparent operations, but with address translation as an additional step in the downstream and upstream directions.

A Non-Transparent PCI bridge is used when there is more than one intelligent entity in a system (*such as* multiple processors). The bridge isolates processor domains by providing a Type 0 Configuration header to each CPU, and allowing data to transfer between the domains using address translation. Therefore, a processor on one bus cannot directly detect devices connected to the other bus.

The PCI 6540 has three Base Address registers (BARs), which allow address translation to occur in the downstream and upstream directions, for a total of six BARs. (Refer to Table 2-1.)

Therefore, it is possible to configure three 32-bit Memory BARs, one 32-bit I/O BAR and two 32-bit Memory BARs, or one 32-bit I/O (or Memory) BAR, plus one 64-bit Memory BAR.

In addition to address translation, the PCI 6540 has other communication mechanisms between the processors and their domains. This includes software-based Doorbell interrupts, hardware-based GPIO interrupts, and a semaphore mechanism-based cross-bridge communication mechanism.

(Refer to Section 6.2, “PCI Configuration Register Address Mapping—Non-Transparent Mode,” and Section 20, “Non-Transparent Mode,” for further details.)

Table 2-1. Non-Transparent Mode Base Address Registers

BAR	Downstream Direction (PCIBARx)	Upstream Direction (PCIUBARx)
I/O or Memory BAR 0 (32 bits)	PCIBAR0; Primary PCI:10h, Secondary PCI:50h	PCIUBAR0; Primary PCI:50h, Secondary PCI:10h
Memory BAR 1 (32 bits)	PCIBAR1; Primary PCI:14h, Secondary PCI:54h	PCIUBAR1; Primary PCI:54h, Secondary PCI:14h
Memory BAR 2 (32 bits) or Memory 1 BAR Upper 32 bits	PCIBAR2; Primary PCI:18h, Secondary PCI:58h	PCIUBAR2; Primary PCI:58h, Secondary PCI:18h

## 3 PIN DESCRIPTION

This section describes the PCI 6540 pins (balls), including pin summary, pull-up and pull-down resistor recommendations, power supply decoupling, and pinout listings.

### 3.1 PIN SUMMARY

Tables 3-4 through 3-14 describe each PCI 6540 pin common to all modes of operation:

- PCI-X Primary Bus Interface
- PCI-X Secondary Bus Interface
- Clock-Related
- Reset
- CompactPCI Hot Swap
- JTAG
- Serial EEPROM Interface
- GPIO
- Miscellaneous
- Power and Ground
- No Connect

The pins listed in Table 3-15 are multiplexed between Transparent and Non-Transparent modes.

For a visual view of the PCI 6540 pinout, refer to Section 26, "Mechanical Specs."

Table 3-1 lists abbreviations used in Section 3 to represent various pin types.

**Table 3-1. Pin Type Abbreviations**

Abbreviation	Pin Type
I	CMOS Input (5V input tolerant, I/O $V_{DD}=3.3V$ )
I/O	CMOS Bi-Directional Input Output (5V input tolerant, I/O $V_{DD}=3.3V$ )
O	CMOS Output
OD	Open Drain
OZ	Output Three-State
PCI	PCI/PCI-X Compliant

**Table 3-1. Pin Type Abbreviations (Continued)**

Abbreviation	Pin Type
PI	PCI-X Input (5V input tolerant, I/O $V_{DD}=3.3V$ )
PO	PCI-X Output
PU	Signal is internally pulled up
STS	PCI-X Sustained Three-State Output, Driven High for One CLK before Float (5V input tolerant, I/O $V_{DD}=3.3V$ )
TS	PCI-X Three-State Bi-Directional (5V input tolerant, I/O $V_{DD}=3.3V$ )

### 3.2 PULL-UP AND PULL-DOWN RESISTOR RECOMMENDATIONS

Pull-up and pull-down resistor values are not critical. With the exception of those mentioned in Section 3.2.1, a 10K-Ohm resistor is recommended unless stated otherwise.

#### 3.2.1 Primary and Secondary PCI Bus Interface Pins

The pins detailed in Table 3-2 are generic primary and secondary PCI interface pins. When producing motherboards, system slot cards, adapter cards, backplanes, and so forth, the termination of these pins should follow the guidelines detailed in *PCI r2.3* and *PCI-X r1.0b*.

**Table 3-2. Generic PCI Bus Interface Pins that follow *PCI r2.3* and *PCI-X r1.0b* Layout Guidelines**

Bus	Pin Name
Primary	P_ACK64#, P_AD[63:0], P_CBE[7:0]#, P_DEVSEL#, P_FRAME#, P_GNT#, P_IDSEL, P_INTA#, P_IRDY#, P_LOCK#, P_M66EN, P_PAR, P_PAR64, P_PERR#, P_REQ#, P_REQ64#, P_SERR#, P_STOP#, P_TRDY#
Secondary	S_ACK64#, S_AD[63:0], S_CBE[7:0]#, S_DEVSEL#, S_FRAME#, S_GNT[7:0]#, S_IDSEL, S_INTA#, S_IRDY#, S_LOCK#, S_M66EN, S_PAR, S_PAR64, S_PERR#, S_REQ[7:0]#, S_REQ64#, S_SERR#, S_STOP#, S_TRDY#

The following guidelines for these pins are not exhaustive and should be read in conjunction with the appropriate sections of *PCI r2.3* and *PCI-X r1.0b*.

PCI control signals require a pull-up resistor on the motherboard to ensure that these signals are always at valid values when a PCI Bus agent is not driving the bus. These control signals include ACK64#, DEVSEL#, FRAME#, INTA#, IRDY#, LOCK#, PERR#, REQ64#, SERR#, STOP#, and TRDY#. The 32-bit point-to-point and shared bus signals do **not** require pull-up resistors, as bus parking ensures that these signals remain stable. The other 64-bit signals—AD[63:32], CBE[7:4]# and PAR64—also require pull-up resistors, as these signals are not driven during 32-bit transactions. Depending on the application, M66EN may also require a pull-up resistor.

The value of these pull-up resistors depends on the bus loading. *PCI r2.3* provides formulas for calculating these resistors.

When making adapter card devices where the PCI 6540 primary port is wired to the PCI connector, pull-up resistors are not required because they are pre-installed on the motherboard.

Based on the above, in an embedded design, pull-up resistors may be required for PCI control signals on the primary and secondary buses. Whereas, for a PCI adapter card design, pull-up resistors are required only on the PCI 6540 port that is not connected to the motherboard or host system.

S\_REQ[7:1]# inputs must be pulled high with a 10K-Ohm pull-up resistor. A 10K-Ohm pull-up resistor is also required for S\_REQ0# if S\_CFN#=0.

Pull S\_GNT[7:1]# high if the PCI 6540 is used in Universal Transparent mode or if S\_CFN#=1.

### 3.2.2 Primary and Secondary Clock Pins

Clock routing is detailed in Section 4, "Clocking." Pull-up resistors are not required on the S\_CLKO[4:0] pins; however, a series termination resistor is required when using these pins. S\_CLKO0 may require a pull-up resistor when this pin is used as a result of S\_CLKOFF=1, or disabled (Transparent mode—CLKCNTRL[1:0]=11b; PCI:68h, Non-Transparent mode—CLKCNTRL[1:0]=11b; PCI:94h). Table 3-3

delineates the remaining clock pin resistor requirements.

**Table 3-3. Clock Pin Pull-Up/Pull-Down Resistor Requirements**

Resistor Requirements	Pin Name
Must pull low	P_CLKOE, P_CR, S_CR
Pull low or connect to 3.3V power supply	P_PLEN#, S_CLKIN_STB, S_PLEN#
Pull high or low if unused	OSCIN, REFCLK
Optionally pull high or low	MSK_IN, OSCSEL#, S_CLKOFF
Pull-up or pull-down resistor not required	P_CLKIN, S_CLKIN, S_CLKO[4:1]*

**Note:** \* Refer also to the text preceding this table.

### 3.2.3 Reset Pins

The P\_RSTIN# and S\_RSTIN# Reset signals may require a pull-up resistor, depending on the application.

The P\_RSTOUT# and S\_RSTOUT# Reset signals do not require pull-up nor pull-down resistors.

**Note:** PWRGD requires a clean, low-to-high transition. This input is not internally de-bounced. Pull-up this input to 3.3V, rather than 5V.

### 3.2.4 CompactPCI Hot Swap Pins

Pull the EJECT pin low if unused.

The ENUM# pin, if used, requires a pull-up resistor.

The L\_STAT CompactPCI Hot Swap signal, if used, does not require a pull-up nor pull-down resistor. L\_STAT must be pulled high if unused.

### 3.2.5 JTAG/Boundary Scan Pins

The TCK, TDI, and TMS JTAG signals must be pulled high or low to a known state, using an external resistor. TRST# must be pulled low, using a 330 Ohm resistor.

The TDO signal does not require a pull-up nor pull-down resistor.

### 3.2.6 Serial EEPROM Pins

EEPCLK does not require a pull-up nor pull-down resistor. EEPDATA requires an external pull-up resistor.

### 3.2.7 GPIO Pins

When programmed as outputs, the GPIO pins do not require external pull-up nor pull-down resistors. If configured as inputs, pull the GPIO pins high or low, depending on the application.

### 3.2.8 Miscellaneous Pins

The BPCC\_EN, DEV64#, P\_BOOT, TRANS#, and U\_MODE signals may optionally be pulled high or low. S\_CFN# may also optionally be pulled high or low, but must be tied low to use the Internal Arbiter.

P\_XCAP is normally configured by the host. In an embedded system, the P\_XCAP pin may be pulled high or low. When P\_XCAP is pulled high, the primary port is forced to use PCI-X Bus protocol. In general, pull the P\_XCAP pin low. When P\_XCAP is pulled low, the primary port bus protocol is set by the PCI-X Initialization Pattern when P\_RSTIN# is de-asserted.

In Transparent mode, tie S\_XCAP\_PU to S\_XCAP\_IN by way of a 1K-Ohm resistor. S\_XCAP\_IN should follow the Programmable Pull-Up and Binary Input method, as defined in *PCI-X r1.0b*, "Detection of PCI-X Add-in Card Capability" appendix. *PCI-X r1.0b* allows detection of 66 MHz Conventional PCI, PCI-X 100 MHz (with GPIO13=1), or PCI-X 133 MHz (with GPIO13=0) adapter cards or devices on the secondary interface.

In Non-Transparent mode, pull S\_XCAP\_IN low.

Provide P\_TST[1:0] and S\_TST[1:0] with the option of being pulled high or low. TEST **must** be pulled low.

Externally pull the ReserveIO[2:1] pins high.

In Transparent mode, pull P\_PME# and S\_PME# high, as they are not used. For Transparent mode applications that require the PME# function, directly connect P\_PME# to S\_PME#, bypassing the PCI 6540.

In Non-Transparent applications, connect P\_PME# and S\_PME# directly to the primary or secondary port PCI connector. Pull these signals high if unused.

### 3.2.9 Multiplexed Transparent and Non-Transparent Pins

Pull PRV\_DEV (Transparent mode) or XB\_MEM (Non-Transparent mode) high or low.

For P\_INTA# and S\_INTA# resistor requirements, refer to Table 3-2.

## 3.3 POWER SUPPLY DECOUPLING

Decouple all  $V_{DD\_CORE}$ ,  $V_{DD\_IO}$ ,  $P\_AV_{DD}$  and  $S\_AV_{DD}$  lines. The decoupling level depends on power plane routing and the acceptable level of supply noise. In an ideal case, decouple all the above power supply pins, using a parallel combination of 100 nF and 10 nF capacitors. Use of the 10 nF capacitors is due to the relatively high inductance of 100 nF capacitors, which can prevent the capture of fast transients.

Due to routing constraints, it may not be possible to add the parallel combination to all supply pins. In this case, the 10 and 100 nF capacitors can be used alternately among the supply pins.

Low-inductance 100 nF capacitors are available, which may be used in place of the 100 nF/10 nF parallel combination.

Take care when choosing the capacitor material. Some types have poor thermal characteristics, resulting in the capacitance value substantially dropping at higher temperatures.

Connect decoupling capacitors to the appropriate ground plane. **Do not** decouple digital supplies to the clean analog Phase-Locked Loop (PLL) grounds or vice versa.

Phase-locked loops are sensitive to power and ground noise. Using the same supply/ground for more than one PLL, or the digital supply/ground (Core or I/O Ring) for either PLL is not recommended, as noise coupling into the PLL supply/ground can cause PLL malfunction.

Each PLL requires a dedicated and independent power supply and ground. Ideally, decouple each PLL supply with 100 pF, 47 nF and 10  $\mu$ F capacitors, in parallel. One set of capacitors is required for each PLL supply.

In addition to the above, a 10  $\mu$ F bulk decoupling capacitor for the digital supply is also recommended. The number and placement of this capacitor depends on the power supply and board design.

### 3.4 PINOUT COMMON TO ALL OPERATING MODES

**Note:** Refer to Section 3.2 for pull-up and pull-down resistor recommendations not specifically stated in these tables.

**Table 3-4. Primary PCI/PCI-X Bus Interface Pins**

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
P_ACK64#	Primary 64-Bit Transfer Acknowledge	1	I/O STS PCI	W14	When asserted by the target device, indicates that the target can perform 64-bit Data transfers. Uses the same timing as P_DEVSEL#. When de-asserting, driven high for one cycle before being placed into a high-impedance state.
P_AD[31:0]	Primary Address and Data, Lower 32 Bits	32	I/O TS PCI	R4, R2, R1, T4, T3, T2, T1, U4, U1, V2, V1, W2, W1, V7, W7, Y7, W10, Y10, T11, U11, V11, W11, Y11, T12, W12, Y12, U13, V13, W13, Y13, U14, V14	Multiplexed Address and Data Bus. P_AD[31:0] provide the lower 32 Address and/or Data pins. Address is indicated by P_FRAME# assertion during PCI and PCI-X transactions. 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 P_IRDY# and P_TRDY# are asserted. During bus idle, the PCI 6540 drives P_AD[31:0] to valid logic levels when P_GNT# is asserted. Additionally, these lines provide a portion of the attribute during the Attribute phase of PCI-X transactions. (Refer to Section 14, "PCI Bus Arbitration," for further details.)
P_AD[63:32]	Primary Address and Data, Upper 32 Bits	32	I/O TS PCI	V16, W16, Y16, U17, V17, W17, Y17, Y18, W18, U18, Y19, W19, V19, W20, V20, U19, U20, T17, T18, T19, T20, R17, R19, R20, P17, P18, P19, P20, N17, N18, N19, N20	Multiplexed Address and Data Bus. P_AD[63:32] provide the upper 32 Address and/or Data pins. During an Address phase (when using the DAC command and P_REQ64# is asserted), the upper 32 bits of a 64-bit address are transferred; otherwise, these bits are undefined. During a Data phase, the upper 32 bits of data are transferred if a 64-bit transaction is negotiated by P_REQ64# and P_ACK64# assertion.

Table 3-4. Primary PCI/PCI-X Bus Interface Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
P_CBE[3:0]#	Primary Lower Command and Byte Enables	4	I/O TS PCI	U3, U8, V10, U12	Multiplexed Command and Byte Enable fields. Provides the transaction type during the PCI and PCI-X Address phase. In the Data phase of PCI and PCI-X Memory Write transactions, P_CBE[3:0]# provide Byte Enables. During the PCI-X Attribute phase, these signals provide a portion of the attribute information. During bus idle, the PCI 6540 drives P_CBE[3:0]# to valid logic levels when P_GNT# is asserted.
P_CBE[7:4]#	Primary Upper Command and Byte Enables	4	I/O TS PCI	U15, W15, Y15, U16	Multiplexed Command and Byte Enable fields. During an Address phase (when using the DAC command and P_REQ64# is asserted), the bus command is transferred on these pins; otherwise, P_CBE[7:4]# are <b>reserved</b> and indeterminate. If a 64-bit transaction is negotiated by P_REQ64# and P_ACK64# assertion, then during a PCI and PCI-X Memory Write transaction Data phase, these pins indicate which byte lanes carry meaningful data.
P_DEVSEL#	Primary Device Select	1	I/O STS PCI	T9	Asserted by the target, indicating that the device is accepting the transaction. As a master, the PCI 6540 waits for P_DEVSEL# assertion within five cycles of P_FRAME# assertion; otherwise, the transaction terminates with a Master Abort. Before being placed into a high-impedance state, P_DEVSEL# is driven to a high state for one cycle.
P_FRAME#	Primary Frame	1	I/O STS PCI	V8	Driven by the initiator of a transaction to indicate the beginning and duration of an access. P_FRAME# de-assertion indicates the final Data phase requested by the initiator. Before being placed into a high-impedance state, P_FRAME# is driven to a high state for one cycle.
P_GNT#	Primary Grant	1	PI	P4	When asserted, the PCI 6540 can access the primary bus. During bus idle with P_GNT# asserted, the PCI 6540 drives P_ADx, P_CBEx#, P_PAR, and P_PAR64 to valid logic levels.
P_IDSEL	Primary Initialization Device Select	1	PI	U2	Used as a Chip Select line for Type 0 Configuration accesses to PCI 6540 Configuration space.
P_IRDY#	Primary Initiator Ready	1	I/O STS PCI	W8	Driven by the initiator of a transaction to indicate its ability to complete the current Data phase on the primary bus. Before being placed into a high-impedance state, P_IRDY# is driven to a de-asserted state for one cycle.



Table 3-4. Primary PCI/PCI-X Bus Interface Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
P_LOCK#	Primary Lock	1	I/O STS PCI	W9	Asserted by the bus master, indicating an atomic operation that may require multiple transactions to complete.
P_M66EN	Primary 66 MHz Enable	1	PI	M20	Set high to allow 66 MHz primary bus operation. Along with S_M66EN, controls the frequency output to the S_CLKO[4:0] pins. (Refer to Section 4, "Clocking," for further details.)
P_PAR	Primary Parity, Lower 32 Bits	1	I/O TS PCI	U10	Parity is even across P_AD[31:0], P_CBE[3:0]#, and P_PAR [such as, an even number of ones (1)]. P_PAR is an input and valid and stable for one cycle after the Address phase (indicated by P_FRAME# assertion) for address parity. During Read transactions, P_PAR is an input and valid for one clock after P_IRDY# assertion. During Write transactions, P_PAR is an output and valid for one clock after P_TRDY# assertion. P_PAR is placed into a high-impedance state one cycle after the P_AD[31:0] lines are placed into a high-impedance state. During bus idle, the PCI 6540 drives P_PAR to a valid logic level when P_GNT# is asserted.
P_PAR64	Primary Parity, Upper 32 Bits	1	I/O TS PCI	M16	Even parity across P_AD[63:32] and P_CBE[7:4]#. P_PAR64 must be valid for one clock after each Address phase on transactions in which P_REQ64# is asserted. For Data phases, after P_PAR64 is valid, P_PAR64 remains valid until one Clock cycle after the current Data phase completes.
P_PERR#	Primary Parity Error	1	I/O STS PCI	Y9	Asserted when a Data Parity error is detected for data received on the primary interface. Before being placed into a high-impedance state, P_PERR# is driven to a de-asserted state for one cycle.
P_REQ#	Primary Request	1	OZ	P1	Asserted by the PCI 6540 to request ownership of the primary bus to perform a transaction. The PCI 6540 de-asserts P_REQ# for at least two PCI Clock cycles before re-asserting it.
P_REQ64#	Primary 64-Bit Transfer Request	1	I/O STS PCI	Y14	Asserted with P_FRAME# by a PCI Bus master to request a 64-bit Data transfer. The PCI 6540 ignores this input during reset. The PCI 6540 asserts P_REQ64# when the secondary PCI master performs a 64-bit transfer or the FORCE64 option is enabled (MSCOPT[15 and/or 11]=1; PCI:46h).



Table 3-4. Primary PCI/PCI-X Bus Interface Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
P_SERR#	Primary System Error	1	Transparent Mode: OD Non-Transparent Mode: If P_BOOT=0 OD otherwise PI	T10	SERR# can be driven low by any device to indicate a System error condition. The PCI 6540 drives P_SERR# during Transparent mode, or Non-Transparent mode with P_BOOT=0, if one of the following conditions is met: <ul style="list-style-type: none"> <li>Address Parity error</li> <li>Posted Write Data Parity error on target bus</li> <li>S_SERR# is asserted</li> <li>Master Abort during Posted Write transaction</li> <li>Target Abort during Posted Write transaction</li> <li>Posted Write transaction discarded</li> <li>Delayed Write request discarded</li> <li>Delayed Read request discarded</li> <li>Delayed transaction Master Timeout</li> </ul> In Non-Transparent mode with P_BOOT=1, pull P_SERR# high with an external resistor.
P_STOP#	Primary Stop	1	I/O STS PCI	U9	Asserted by the target to end the transaction on the current Data phase. Before being placed into a high-impedance state, P_STOP# is driven to a de-asserted state for one cycle.
P_TRDY#	Primary Target Ready	1	I/O STS PCI	Y8	Driven by the target of a transaction to indicate its ability to complete the current Data phase on the primary bus. Before being placed into a high-impedance state, P_TRDY# is driven to a de-asserted state for one cycle.
<b>Total</b>		88			

Table 3-5. Secondary PCI/PCI-X Bus Interface Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
S_ACK64#	Secondary 64-Bit Transfer Acknowledge	1	I/O STS PCI	B17	When asserted by the target device, indicates that the target can perform 64-bit Data transfers. Uses the same timing as S_DEVSEL#. When de-asserting, driven high for one cycle before being placed into a high-impedance state.
S_AD[31:0]	Secondary Address and Data, Lower 32 Bits	32	I/O TS PCI	A6, B6, C6, D6, A7, B7, C7, D7, C8, D8, A9, B9, D9, E9, A10, B10, E12, A13, B13, C13, D13, A14, B14, C14, A15, B15, D15, A16, B16, C16, D16, A17	Multiplexed Address and Data Bus. S_AD[31:0] provide the lower 32 Address and/or Data pins. Address is indicated by S_FRAME# assertion during PCI and PCI-X transactions. 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 S_IRDY# and S_TRDY# are asserted. During bus idle, the PCI 6540 drives S_AD[31:0] to valid logic levels when the PCI 6540 is granted secondary bus ownership. Additionally, these lines provide a portion of the attribute during the Attribute phase of PCI-X transactions. (Refer to Section 14, "PCI Bus Arbitration," for further details.)
S_AD[63:32]	Secondary Address and Data, Upper 32 Bits	32	I/O TS PCI	B20, C20, C19, D20, D19, D18, D17, E20, E19, E18, E17, F20, F19, F17, G20, G19, G18, G17, H20, H19, H18, H17, J20, J19, J17, J16, K20, K19, K18, K17, K16, L20	Multiplexed Address and Data Bus. S_AD[63:32] provide the upper 32 Address and/or Data pins. During an Address phase (when using the DAC command and S_REQ64# is asserted), the upper 32 bits of a 64-bit address are transferred; otherwise, these bits are undefined. During a Data phase, the upper 32 bits of data are transferred if a 64-bit transaction is negotiated by S_REQ64# and S_ACK64# assertion.
S_CBE[3:0]#	Secondary Lower Command and Byte Enables	4	I/O TS PCI	B8, C10, D12, D14	Multiplexed Command and Byte Enable fields. Provides the transaction type during the PCI and/or PCI-X Address phase. In the Data phase of PCI and PCI-X Memory Write transactions, S_CBE[3:0]# provide the Byte Enables. During the PCI-X Attribute phase, these signals provide a portion of the attribute information. During bus idle, the PCI 6540 drives S_CBE[3:0]# to valid logic levels when the PCI 6540 is granted secondary bus ownership. (Refer to Section 14, "PCI Bus Arbitration," for further details.)

Table 3-5. Secondary PCI/PCI-X Bus Interface Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
S_CBE[7:4]#	Secondary Upper Command and Byte Enables	4	I/O TS PCI	A18, B18, A19, B19	Multiplexed Command and Byte Enable fields. During an Address phase (when using the DAC command and S_REQ64# is asserted), the bus command is transferred on these pins; otherwise, S_CBE[7:4]# are <b>reserved</b> and indeterminate. If a 64-bit transaction is negotiated by S_REQ64# and S_ACK64# assertion, then during a PCI and PCI-X Memory Write transaction Data phase, these pins indicate which byte lanes carry meaningful data.
S_DEVSEL#	Secondary Device Select	1	I/O STS PCI	B11	Asserted by the target, indicating that the device is accepting the transaction. As a master, the PCI 6540 waits for the S_DEVSEL# assertion within five cycles of S_FRAME# assertion; otherwise, the transaction terminates with a Master Abort. Before being placed into a high-impedance state, S_DEVSEL# is driven to a high state for one cycle.
S_FRAME#	Secondary Frame	1	I/O STS PCI	D10	Driven by the initiator of a transaction to indicate the beginning and duration of an access. S_FRAME# de-assertion indicates the final Data phase requested by the initiator. Before being placed into a high-impedance state, S_FRAME# is driven to a high state for one cycle.
S_GNT0#	Secondary Grant 0	1	Transparent and Non-Transparent Modes: If S_CFN#=0 OZ otherwise PI  Universal Non-Transparent Mode: PI	E1	Asserted by the PCI 6540 to grant the secondary bus to a secondary bus master when using internal arbitration (S_CFN#=0). When external arbitration is active (S_CFN#=1), S_GNT0# becomes the external bus Grant input to the PCI 6540. In Universal Non-Transparent mode, S_GNT0# becomes the PCI 6540 secondary port Grant input. During bus idle with S_GNT0# asserted, the PCI 6540 drives S_ADx, S_CBEx#, S_PAR, and S_PAR64 to valid logic levels.
S_GNT[7:1]#	Secondary Internal Arbiter Grant 7 to 1	7	Transparent and Non-Transparent Modes: If S_CFN#=0 OZ otherwise PI  Universal Non-Transparent Mode: PI	G2, G3, G4, F1, F2, F3, F4	Asserted by the PCI 6540 to grant the secondary bus to a secondary bus master. The PCI 6540 de-asserts S_GNT[7:1]# for at least two PCI Clock cycles before re-asserting them. Pull S_GNT[7:1]# high if the PCI 6540 is used in Universal Transparent mode or if S_CFN#=1. <b>Note:</b> S_GNT[7:1]# are not used in Non-Transparent mode.

Table 3-5. Secondary PCI/PCI-X Bus Interface Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
S_IDSEL	Secondary Initialization Device Select (Non-Transparent Mode)	1	PI	A8	Valid only in Non-Transparent mode. Used as a Chip Select line for Type 0 Configuration accesses to the PCI 6540 secondary Configuration space. If not used, pull S_IDSEL low.
S_IRDY#	Secondary Initiator Ready	1	I/O STS PCI	E10	Driven by the initiator of a transaction to indicate its ability to complete the current Data phase on the secondary bus. Before being placed into a high-impedance state, S_IRDY# is driven to a de-asserted state for one cycle.
S_LOCK#	Secondary Lock	1	I/O STS PCI	D11	Asserted by the bus master, indicating an atomic operation that may require multiple transactions to complete.
S_M66EN	Secondary 66 MHz Enable	1	If P_M66EN=0 OD otherwise PI	L16	Driven low if P_M66EN is low; otherwise, driven from outside to select 66 or 33 MHz. S_M66EN <b>must</b> be pulled high with a 10K-Ohm resistor. Along with P_M66EN, controls the frequency output to the S_CLKO[4:0] pins. (Refer to Section 4, "Clocking," for further details.)
S_PAR	Secondary Parity, Lower 32 Bits	1	I/O TS PCI	B12	Parity is even across S_AD[31:0], S_CBE[3:0]#, and S_PAR [ <i>such as</i> , an even number of ones (1)]. S_PAR is an input and valid and stable one cycle after the Address phase (indicated by S_FRAME# assertion) for address parity. During Read transactions, S_PAR is an input and valid for one clock after S_IRDY# assertion. During Write transactions, S_PAR is an output and valid for one clock after S_TRDY# assertion. S_PAR is placed into a high-impedance state one cycle after the S_AD[31:0] lines are placed into a high-impedance state. During bus idle, the PCI 6540 drives S_PAR to a valid logic level when the PCI 6540 is granted secondary bus ownership. (Refer to Section 14, "PCI Bus Arbitration," for further details.)
S_PAR64	Secondary Parity, Upper 32 Bits	1	I/O TS PCI	L19	Even parity across S_AD[63:32] and S_CBE[7:4]#. S_PAR64 must be valid for one clock after each Address phase on transactions in which S_REQ64# is asserted. For Data phases, after S_PAR64 is valid, S_PAR64 remains valid until one Clock cycle after the current Data phase completes.

Table 3-5. Secondary PCI/PCI-X Bus Interface Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
S_PERR#	Secondary Parity Error	1	I/O STS PCI	E11	Asserted when a Data Parity error is detected for data received on the secondary interface. Before being placed into a high-impedance state, S_PERR# is driven to a de-asserted state for one cycle.
S_REQ0#	Secondary Request 0	1	Transparent and Non-Transparent Modes: If S_CFN#=0 PI otherwise OZ  Universal Non-Transparent Mode: OZ	B1	When the Internal PCI Arbiter is enabled (S_CFN#=0), S_REQ0# is asserted by an external device to request secondary bus ownership to perform a transaction. When using external arbitration (S_CFN#=1 or Universal Non-Transparent mode), S_REQ0# becomes the External Request output from the PCI 6540. In Universal Non-Transparent mode, S_REQ0# becomes the PCI 6540 secondary port Request output. S_REQ0# <b>must</b> be externally pulled up through a 10K-Ohm resistor if S_CFN#=0.
S_REQ[7:1]#	Secondary Internal Arbiter Request 7 to 1	7	PI	E3, E4, D1, D2, D3, C1, C2	Asserted by an external device to request secondary bus ownership to perform a transaction. S_REQ[7:1]# <b>must</b> be externally pulled up through 10K-Ohm resistors, including those pins which are not connected to other bus master devices. S_REQ[7:1]# are not used in Universal Non-Transparent mode and when S_CFN#=1.
S_REQ64#	Secondary 64-Bit Transfer Request	1	I/O STS PCI	C17	Asserted with S_FRAME# by a secondary PCI Bus master to request a 64-bit Data transfer. The PCI 6540 asserts S_REQ64# low during reset and when the primary PCI master performs a 64-bit transfer or the FORCE64 option is enabled (MSCOPT[15 and/or 11]=1; PCI:46h).

Table 3-5. Secondary PCI/PCI-X Bus Interface Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
S_SERR#	Secondary System Error	1	Transparent Mode: PI  Non-Transparent Mode: If P_BOOT=0 PI otherwise OD	A12	SERR# can be driven low by any device to indicate a System error condition. The PCI 6540 drives S_SERR# only during Non-Transparent mode with P_BOOT=1 and if the following occurs: <ul style="list-style-type: none"> <li>Address Parity error</li> <li>Posted Write Data Parity error on target bus</li> <li>P_SERR# is asserted</li> <li>Master Abort during Posted Write transaction</li> <li>Target Abort during Posted Write transaction</li> <li>Posted Write transaction discarded</li> <li>Delayed Write request discarded</li> <li>Delayed Read request discarded</li> <li>Delayed transaction Master Timeout</li> </ul> Pull S_SERR# high with an external resistor.
S_STOP#	Secondary Stop	1	I/O STS PCI	C11	Asserted by the secondary target to end the transaction on the current Data phase. Before being placed into a high-impedance state, S_STOP# is driven to a de-asserted state for one cycle.
S_TRDY#	Secondary Target Ready	1	I/O STS PCI	A11	Driven by the target of a transaction to indicate its ability to complete the current Data phase on the secondary bus. Before being placed into a high-impedance state, S_TRDY# is driven to a de-asserted state for one cycle.
<b>Total</b>		102			

Table 3-6. Clock-Related Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
MSK_IN	Mask	1	PI	J1	Used with GPIO[2, 0] to shift in a serial stream of bits to the Clock Control register (Transparent mode—CLKCNTRL; PCI:68h, Non-Transparent mode—CLKCNTRL; PCI:94h) to enable or disable the S_CLKO[4:0] Clock Output buffers during reset. MSK_IN can be pulled low to enable, or high to disable, all S_CLKO[4:0] buffers. <b>Note:</b> S_CLKOFF can also be used to enable or disable S_CLKO[4:0].
OSCIN	External Oscillator Input	1	PI	P3	External clock input used to generate secondary output clocks when enabled through the OSCSEL# pin. Pull high or low if unused.
OSCSEL#	External Oscillator Enable	1	PI	N1	Enables external clock connection for the secondary interface. If low, the secondary bus clock outputs use the clock signal from OSCIN, instead of P_CLKIN, to generate S_CLKO[4:0]. May optionally be pulled high or low.
P_CLKIN	Primary PCI Clock Input	1	PI	N4	Provides timing for primary interface transactions.
P_CLKOE	Primary Clock Output Enable	1	PI	R5	Test pin. <b>Must</b> be pulled low for normal operation. Values: 0 = Disables Test function. 1 = S_CLKO3 is for primary PLL test and S_CLKO4 is for secondary PLL test.
P_CR	Primary PLL Range Control	1	PI	T6	Selects the primary PLL operating range. <b>Must</b> be pulled low for normal operation. Pull or tie to V <sub>SS</sub> .
P_PLEN#	Primary PLL Enable	1	PI	T7	Values: 0 = Enables primary PLL. 1 = Disables primary PLL. <b>Must</b> be pulled low or connected to 3.3V power supply.
REFCLK	Reference Clock Input	1	PI	P2	When used, REFCLK should be a fixed frequency input (14.318 MHz recommended), which is used by the internal counters to determine the primary and secondary PCI clock frequency. Pull high or low if unused.

Table 3-6. Clock-Related Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
S_CLKIN	Secondary PCI Clock input	1	PI	J5	Provides timing for all transactions on the secondary interface <b>except</b> during Universal Non-Transparent mode (U_MODE=1 and TRANS#=1). In Universal Non-Transparent mode, this input is ignored by the PCI 6540 and internal logic uses the input clock from the S_CLKO0 pin.
S_CLKIN_STB	Secondary Clock Input Stable	1	PI	K5	Values: 0 = S_RSTOUT# remains asserted until S_CLKIN_STB is 1. 1 = Indicates external secondary Clock PLL and external S_CLKIN are stable. <b>Must</b> be pulled low or connected to 3.3V power supply. If not used, connect to a 3.3V power supply.
S_CLKO0	Secondary Clock 0	1	Transparent and Non-Transparent Modes: If S_CLKOFF=0 PO otherwise PI  Universal Non-Transparent Mode: PI	K3	In Transparent mode, S_CLKO0 is used as a Clock Output buffer, which is derived from the P_CLKIN or OSCIN (if OSCSEL# is low) clock; however, phase synchronization is not guaranteed. This clock can be placed into a high-impedance state, using the S_CLKOFF pin. In Universal Non-Transparent mode (U_MODE=1 and TRANS#=1), S_CLKO0 becomes clock input to the secondary interface. S_CLKIN is ignored in this mode. This is helpful when producing cards for use in peripheral and system slots in CompactPCI systems. When configured in a peripheral card, allows S_CLKO0 to be a clock input from the CLK signal of a CompactPCI backplane. Therefore, when used in a CompactPCI system slot (PCI 6540 is operating in Transparent mode), S_CLKO0 drives the CompactPCI backplane, and when used in a Peripheral slot (PCI 6540 is operating in Non-Transparent mode), the backplane clock drives S_CLKO0. Pull-up resistors are not required on S_CLKO0; however, a series termination resistor is required when using this pin. A pull-up resistor may be required when S_CLKO0 is used as a result of S_CLKOFF=1, or disabled (Transparent mode—CLKCNTRL[1:0]=11b; PCI:68h, Non-Transparent mode—CLKCNTRL[1:0]=11b; PCI:94h). <b>Note:</b> Output clocks are not recommended for PCI-X applications. External high-quality buffers are recommended.



Table 3-6. Clock-Related Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
S_CLKO[4:1]	Secondary Clock Output 4 to 1	4	OZ	L4, L5, K1, K2	<p>Provides P_CLKIN or OSCIN (if enabled) frequency output clocks; however, phase synchronization is not guaranteed. These clocks can be set to drive 0, using S_CLKOFF.</p> <p>When the S_CLKOFF pin is low, the associated Clock Control register Disable bit (Transparent mode—CLKCNTRL[8:0]; PCI:68h, Non-Transparent mode—CLKCNTRL[8:0]; PCI:94h) places the associated S_CLKOx pin into a high-impedance state. This function can be used when the disabled clock buffer does not connect to any device.</p> <p>Pull-up resistors are not required on S_CLKO[4:1]; however, a series termination resistor is required when using these pins.</p> <p><b>Note:</b> Output clocks are not recommended for PCI-X applications. External high-quality buffers are recommended.</p>
S_CLKOFF	Secondary Clock Output Disable	1	PI	K4	<p>Values:</p> <p>0 = Enables S_CLKO[4:0] output. This enable can be overridden by the Clock Control register Disable bits (Transparent mode—CLKCNTRL[8:0]; PCI:68h, Non-Transparent mode—CLKCNTRL[8:0]; PCI:94h).</p> <p>1 = S_CLKO[4:1] output are driven low and S_CLKO0 is placed into a high-impedance state. This disable <b>cannot</b> be overridden by the Clock Control register Disable bits.</p> <p>May optionally be pulled high or low.</p>
S_CR	Secondary PLL Range Control	1	PI	F5	Selects the secondary PLL operating range. <b>Must</b> be pulled low for normal operation. Pull or tie to V <sub>SS</sub> .
S_PLEN#	Secondary PLL Enable	1	PI	E6	<p>Values:</p> <p>0 = Enables secondary PLL.</p> <p>1 = Disables secondary PLL.</p> <p><b>Must</b> be pulled low or connected to 3.3V power supply.</p>
<b>Total</b>		18			

Table 3-7. Reset Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
P_RSTIN#	Primary Reset Input	1	PI	L2	When asserted, outputs are asynchronously placed into a high-impedance state, P_SERR# and P_GNT# floated, all primary PCI signals ( <i>except</i> P_RSTOUT#) are placed into a high-impedance state, and no bus parking is asserted. All primary port PCI standard Configuration registers at offsets 00h to 3Fh revert to their default state. May require a pull-up resistor, depending on the application.
P_RSTOUT#	Primary Reset Output (Non-Transparent Mode)	1	PO	L1	Valid only in Non-Transparent mode. Asserted when either of the following conditions is met: <ul style="list-style-type: none"> <li>S_RSTIN# is asserted when the primary port has boot priority (P_BOOT=1).</li> <li>Non-Transparent Diagnostic Control register Primary Reset bit in Configuration space is set (DCNTRL[5]=1; PCI:D9h).</li> </ul>
PWRGD	Power Good Input	1	PI	N3	The asserting and de-asserting edges of PWRGD can be asynchronous to P_CLKIN and S_CLKIN.  <b>Important Note:</b> The PCI 6540 requires a clean low-to-high transition PWRGD input. PWRGD is not internally de-bounced—it must be externally de-bounced and a high input must reflect that the power is indeed stable. When this input is low, all PCI 6540 state machines and registers are reset and all outputs, except S_RSTOUT# and P_RSTOUT#, are placed into a high-impedance state. Pull-up this input to 3.3V, rather than 5V.
S_RSTIN#	Secondary Reset Input	1	PI	H3	When asserted, all secondary PCI signals ( <i>except</i> S_RSTOUT#) are placed into a high-impedance state and bus parking is not asserted. In Transparent or Universal Transparent mode, S_RSTIN# is not used and <b>must</b> be pulled high. In these modes, S_RSTOUT# functions as the secondary port Reset Input pin. In Non-Transparent mode, S_RSTIN# assertion causes all secondary port control logic to be reset. Primary port control logic is not affected. In Universal Non-Transparent mode (U_MODE=1 and TRANS#=1), S_RSTIN# is ignored and S_RSTOUT# is used as the equivalent of S_RSTIN#. May require a pull-up resistor, depending on the application.

Table 3-7. Reset Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
S_RSTOUT#	Secondary Reset Output	1	If Universal Non-Transparent Mode: PI otherwise PO	H2	<p>Asserted when either of the following conditions is met:</p> <ul style="list-style-type: none"> <li>P_RSTIN# is asserted</li> <li>Bridge Control register Secondary Reset bit in Configuration space is set (Transparent mode—BCNTRL[6]=1; PCI:3Eh, Non-Transparent mode—BCNTRL[6]=1; PCI:42h Shadow register)</li> </ul> <p>In Transparent or Universal Transparent mode, S_RSTIN# is disabled and S_RSTOUT# functions as the secondary port Reset Input pin.</p> <p>In Universal Non-Transparent mode (U_MODE=1 and TRANS#=1), S_RSTOUT# is disabled and used as the equivalent of S_RSTIN#.</p>
<b>Total</b>		5			

Table 3-8. CompactPCI Hot Swap Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
ENUM#	Enumeration	1	OD	Y6	<p>Indicates an open-drain based signal asserted when an adapter was inserted or is ready to be removed from a PCI slot. Asserted through the Hot Swap registers (HS_CNTL; PCI:E4h, HS_CSR; PCI:E6h, HS_NEXT; PCI:E5h, and HSSRRC; PCI:9Ch).</p> <p>If used, ENUM# requires a pull-up resistor.</p>
L_STAT	CompactPCI LED On	1	OZ	W6	<p>Indicates the software connection process status.</p> <p>If not used, L_STAT <b>must</b> be pulled high.</p>
EJECT	Hot Swap Eject	1	PI	U6	<p>Used to detect the insertion of a Hot Swap device. When asserted, the PCI 6540 asserts ENUM#. An external pull-down resistor is recommended.</p> <p>If not used, EJECT <b>must</b> be at logic 0 and pulled low.</p>
<b>Total</b>		3			

Table 3-9. JTAG/Boundary Scan Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
TCK	Test Clock Input	1	I PU	Y2	Used to clock state information and test data into and out of the PCI 6540 during Test Access Port (TAP) operation. Pull TCK high or low to a known state, using an external resistor.
TDI	Test Data Input	1	I PU	V4	Used to serially shift test data and test instructions into the PCI 6540 during TAP operation. Pull TDI high or low to a known state, using an external resistor.
TDO	Test Data Output	1	O	W3	Used to transmit serial data from the PCI 6540 TAP. Data from the selected Shift register is shifted out of TDO.
TMS	Test Mode Select	1	I PU	U5	Used to control the PCI 6540 TAP controller state. Pull TMS high or low to a known state, using an external resistor.
TRST#	Test Reset	1	I PU	Y3	Provides asynchronous initialization of the TAP controller. TRST# <b>must</b> be pulled low with a 330 Ohm resistor.
<b>Total</b>		5			

**Note:** The JTAG interface is described in Section 24, "Testability/Debug."

Table 3-10. Serial EEPROM Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
EEPCLK	Serial EEPROM Clock	1	PO	Y5	Clock signal to the serial EEPROM interface. Used during autoload and for VPD functions.
EEPDATA	Serial EEPROM Data	1	I/O	Y4	Serial data interface to the serial EEPROM. Requires an external pull-up resistor.
<b>Total</b>		2			

Table 3-11. General Purpose I/O Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
GPIO[3:0]	General Purpose Input/Output 3 to 0	4	I/O PU	M5, M4, M2, M1	General purpose signals, programmable as input-only or bi-directional by writing to the GPIO Output Enable register (Transparent mode—GPIOOE[3:0]; PCI:66h, Non-Transparent mode—GPIOOE[3:0]; PCI:9Ah). During P_RSTIN# assertion, GPIO[2, 0] are used to shift in the Clock Disable serial data. (Refer to Section 15, “GPIO Interface,” for further details.) If configured as input, pull high or low, depending on the application.
GPIO[7:4]	General Purpose Input/Output 7 to 4	4	I/O PU	A5, B5, C5, D5	General purpose signals, programmable as input-only or bi-directional by writing to the GPIO Output Enable register (GPIOOE[7:4]; PCI:9Eh). During Non-Transparent mode: <ul style="list-style-type: none"> <li>GPIO5 can be enabled as an external interrupt source on the primary port to trigger S_INTA#</li> <li>GPIO4 can be enabled as an external interrupt source on the secondary port to trigger P_INTA#</li> </ul> GPIO[7:4] are internally pulled up. (Refer to Section 15, “GPIO Interface,” for further details.)
GPIO[15:14, 12:8]	General Purpose Input/Output 15, 14, and 12 to 8	7	I/O	A2, B2, B3, D4, A4, B4, C4	General purpose signals, programmable as input-only or bi-directional by writing to the GPIO Output Enable register (GPIOOE[15:8][7:6, 4:0]; PCI:A2h). During PWRGD reset, the status of these pins is latched in the Power-Up Status register (PWRUPSR; PCI:A0h) for general user-defined use. (Refer to Section 15, “GPIO Interface,” for further details.) <b>Notes:</b> <i>Recommended use (pull-up to 3.3V):</i> <b>GPIO15</b> —Primary Power State. 1 = Primary port power is stable. <b>GPIO14</b> —Secondary Power State. 1 = Secondary port power is stable.

Table 3-11. General Purpose I/O Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
GPIO13 (as Transparent Mode PCI-X 100 MHz Selection Input)	General Purpose Input/ Output 13	1	I/O	A3	<p>General purpose signal, programmable as input-only or bi-directional by writing to the GPIO Output Enable register (GPIOOE[15:8][5]; PCI:A2h).</p> <p><b>Notes:</b> <i>Recommended use:</i> Transparent mode, 100/133 MHz PCI-X clock frequency.</p> <p>During P_RST# assertion, GPIO13 status is latched in the Power-Up Status register (PWRUPSR[5]; PCI:A0h). This value determines the maximum allowable PCI-X speed status value, reported in the standard PCI-X Secondary Status register (PCIXSSR; PCI:F2h).</p> <p><b>Transparent mode only</b>—pull GPIO13 low for all 133 MHz-capable applications.</p> <p>Transparent mode PCI-X Clock Selection: 0 = Secondary PCI-X maximum speed is 133 MHz. 1 = Secondary PCI-X maximum speed is 100 MHz.</p>
<b>Total</b>		16			

Table 3-12. Miscellaneous Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
BPCC_EN	Bus/Power Clock Control	1	PI	J2	When tied high and the PCI 6540 is placed into the D <sub>3hot</sub> power state, the PCI 6540 places the secondary bus into the B <sub>2</sub> power state. The PCI 6540 disables the secondary clocks and drives them to 0. When pulled low, placing the PCI 6540 into the D <sub>3hot</sub> power state has no effect on the secondary bus clocks.
DEV64#	64-Bit Device	1	PI	W4	The complementary value of this input pin is reflected in PCI-X Bridge Status register (PCIXBSR[16]; PCI:F4h). DEV64# has no effect on bus transactions. Values: 0 = 64-bit bus. 1 = 32-bit bus. May optionally be pulled high or low.
P_BOOT	Primary Port Boot Priority	1	PI	U7	Used in Non-Transparent mode. Values: 0 = Secondary port uses boot priority—the secondary port must set the S_PORT_READY bit before the primary port can proceed to boot. 1 = Primary port uses boot priority—the primary port must set the P_PORT_READY bit before the secondary port can proceed to boot. May optionally be pulled high or low.
P_PME#	Primary Power Management Event	1	Transparent Mode: OD  Non-Transparent Mode: If P_BOOT=0 OD otherwise PI	M19	In Transparent mode, P_PME# is <i>not</i> normally used. Pull P_PME# high and do <i>not</i> connect it to other Power Management-related signals. For Transparent mode applications that require the PME# function, directly connect P_PME# to S_PME#, bypassing the PCI 6540. In Non-Transparent mode, with the primary port having lower boot priority (P_BOOT=0), P_PME# is always an output and reflects the S_PME# input state if PME is enabled by setting PMCSR[8]=1; PCI:E0h. In Non-Transparent applications, connect P_PME# directly to the primary or secondary port PCI connector. Pull high if unused. Used by secondary port devices to wake up the primary port host.
P_TST[1:0]	Primary Test	2	PI	T14, T15	<b>Reserved</b> inputs. Connect P_TST[1:0] to logic 0 or 1 in layout for timing controls. (Refer to the latest reference design information.) Provide P_TST[1:0] with the option of being pulled high or low.

Table 3-12. Miscellaneous Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
P_XCAP	Primary Port PCI-X Enable	1	PI	N2	<p>Values:</p> <p>0 = Primary port protocol is set by standard PCI/PCI-X reset initialization sequencing.</p> <p>1 = Primary port is forced to run PCI-X protocol.</p> <p>If there is no PCI-X initialization pattern issued by the system host bus, the primary port runs on PCI protocol when P_XCAP is pulled low.</p> <p><b>Note:</b> Refer to Section 3.2.8 for resistor requirements.</p>
ReserveIO[2:1]	Reserved I/O	2	I/O	G1, E2	<b>Reserved.</b> Externally pull high.
S_CFN#	Internal Arbiter Enable	1	PI	H4	<p>Values:</p> <p>0 = Uses Internal Arbiter.</p> <p>1 (or Universal Non-Transparent mode) = Uses External Arbiter. S_REQ0# becomes REQ# output to External Arbiter and S_GNT0# becomes External Arbiter GNT# input.</p> <p>In Universal Non-Transparent mode, the PCI 6540 is configured to use an External Arbiter and S_CFN# becomes <i>don't care</i> input.</p> <p>May optionally be pulled high or low; however, S_CFN# <b>must</b> be tied low to use the Internal Arbiter.</p>
S_PME#	Secondary Power Management Event	1	<p>Transparent Mode: OD</p> <p>Non-Transparent Mode: If P_BOOT=0 PI otherwise OD</p>	L17	<p>In Transparent mode, S_PME# is <b>not</b> normally used. Pull S_PME# high and <b>do not</b> connect it to other Power Management-related signals. For Transparent mode applications that require the PME# function, directly connect S_PME# to P_PME#, bypassing the PCI 6540.</p> <p>In Non-Transparent mode, with the secondary port having lower boot priority (P_BOOT=1), S_PME# is always an output and reflects the P_PME# input state if PME is enabled by setting PMCSR[8]=1; PCI:E0h. In Non-Transparent applications, connect S_PME# directly to the primary or secondary port PCI connector. Pull high if unused.</p> <p>Used by secondary port devices to wake up the primary port host.</p>



Table 3-12. Miscellaneous Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
S_TST[1:0]	Secondary Test	2	PI	E14, E15	<b>Reserved</b> inputs. Connect S_TST[1:0] to logic 0 or 1 in layout for timing controls. (Refer to the latest reference design information.) Provide S_TST[1:0] with the option of being pulled high or low.
S_XCAP_IN	Secondary Port PCI-X Enable	1	PI	N5	In Transparent mode, S_XCAP_IN is pulled high with 56K-Ohm resistor and connected to S_XCAP_PU by way of a 1K-Ohm resistor. Together with S_XCAP_PU, S_XCAP_IN detects the secondary interface bus mode. If S_XCAP_IN is always sampled as 1 when sampled by the PCI 6540 during secondary reset, the secondary port must run PCI-X protocol. In Non-Transparent mode, pull S_XCAP_IN low. The secondary interface bus mode is set by a PCI-X Initialization pattern during secondary reset. <b>Note:</b> Refer to Section 3.2.8 for additional resistor requirements.
S_XCAP_PU	Secondary Port PCI-XCAP Pull-Up	1	OZ	H1	Transparent mode. A 1K-Ohm resistor <b>must</b> be placed between S_XCAP_PU and S_XCAP_IN. <b>Note:</b> S_XCAP_PU is <b>not</b> used in Non-Transparent mode.
TRANS#	Transparent Mode	1	PI	V5	In CompactPCI universal bridge applications, TRANS# can be directly connected to the CompactPCI SYSEN# pin. Values: 0 = PCI 6540 is configured as a standard PCI-to-PCI bridge. 1 = PCI 6540 is operating in Non-Transparent mode. May optionally be pulled high or low.
U_MODE	Universal Mode	1	PI	W5	Used with the TRANS# pin in CompactPCI applications. U_MODE allows one CompactPCI card to be used as a system board in a system slot, and as an intelligent subsystem board in a peripheral slot. Value: 1 = PCI 6540 is configured as a universal bridge. May optionally be pulled high or low.
TEST	Test Pin	1	PI	F15	This is a test pin and <b>must</b> be pulled low.
<b>Total</b>		18			

Table 3-13. Power and Ground Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
P_AV <sub>DD</sub>	Primary PLL Power	2	I	P6, R7	Clean +2.5V for primary PLL.
P_AV <sub>SS</sub>	Primary PLL Ground	2	I	P5, R6	Clean ground for primary PLL.
S_AV <sub>DD</sub>	Secondary PLL Power	2	I	F7, G6	Clean +2.5V for secondary PLL.
S_AV <sub>SS</sub>	Secondary PLL Ground	2	I	F6, G5	Clean ground for secondary PLL.
V <sub>DD_CORE</sub>	Core Power	19	I	E8, E13, F8, F13, G7, G14, H5, H6, H15, H16, N6, N15, N16, P7, P14, R8, R13, T8, T13	+2.5V supply for digital core.
V <sub>DD_IO</sub>	I/O Ring Power	35	I	C3, C15, C18, F9, F10, F11, F12, F18, G8, G13, H7, H14, J6, J15, K6, K15, L3, L6, L15, M6, M15, N7, N14, P8, P13, R3, R9, R10, R11, R12, R18, V3, V6, V15, V18	+3.3V for digital I/O buffers.
V <sub>SS</sub>	Ground	48	I	A1, A20, C9, C12, E5, E16, G9, G10, G11, G12, J3, J7, J9, J10, J11, J12, J14, J18, K7, K9, K10, K11, K12, K14, L7, L9, L10, L11, L12, L14, M3, M7, M9, M10, M11, M12, M14, M18, P9, P10, P11, P12, T5, T16, V9, V12, Y1, Y20	Ground for digital core and I/O.
<b>Total</b>		110			

Table 3-14. No Connect Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
NC	No Connect	10	—	E7, F16, F14, G16, G15, P16, P15, R16, R15, R14	No connect pins, which are not to be connected or used as routing channels. May be used in future PCI 6540 revisions.
<b>Total</b>		10			

## 3.5 PINOUT SPECIFIC TO TRANSPARENT AND NON-TRANSPARENT MODES

Table 3-15. Multiplexed Transparent/Non-Transparent Pins

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
PRV_DEV	Private Device and Memory Enable (Transparent Mode)	1	PI	J4	<p>After power-up, the functions set by PRV_DEV can be modified by software, using the Chip Control register (Transparent mode—CCNTRL[3:2]; PCI:40h, Non-Transparent mode—CCNTRL[3:2]; PCI:D8h). <b>Must</b> be pulled high or low.</p> <p><b>PRV_DEV:</b> When set to 1, the PCI 6540 can mask secondary devices using IDSEL connected to S_AD[23:16] as private devices. Any Type 1 Configuration access to these IDSELs is routed to S_AD24. If there is no device on S_AD24, the rerouted Type 1 Configuration cycles are Master Aborted. The PCI 6540 also reserves Private Memory space for the secondary port. The Memory space can be programmed using the Private Memory Base and Limit registers (Base—PVTMBAR; PCI:6Ch and PVTMBARU32; PCI:70h, Limit—PVTMLMT; PCI:6Eh and PVTMLMTU32; PCI:74h). If the limit is smaller than the base, Private Memory space is disabled. The primary port cannot access this Memory space through the bridge and the secondary port does <b>not</b> respond to Memory cycles addressing this Private Memory space.</p>
XB_MEM	Cross-Bridge Memory Window Enable (Non-Transparent Mode)				<p><b>XB_MEM:</b> When set to 1, the PCI 6540 automatically claims 16 MB of Memory space. This allows the boot-up of the Low-Priority Boot port to proceed without waiting for the Priority Boot port to program the corresponding Memory Base Address registers (BARs). Although the default claims 16 MB, the BARs can be modified by serial EEPROM or software to change the window size. If XB_MEM=1, the P_PORT_READY or S_PORT_READY mechanism is <b>not</b> relevant. Also, if XB_MEM=1, the PCI 6540 autoloads serial EEPROM data up to Group 5 instead of Group 4.</p>

Table 3-15. Multiplexed Transparent/Non-Transparent Pins (Continued)

Symbol	Signal Name	Total Pins	Pin Type	Pin Number	Function
P_CLKRUN#	Primary Clock Run (Transparent Mode)	1	I/O	M17	P_CLKRUN#: Valid only in Transparent mode. Used by the <i>central resource</i> (bus support functions supplied by the host system) to slow down or stop the PCI clock when the clock is enabled.
P_INTA#	Primary Interrupt A# Output (Non-Transparent Mode)		OD		P_INTA#: In Non-Transparent mode, becomes P_INTA# output. Driven by the PCI 6540 to generate a PCI Interrupt request on the primary bus. In Universal Non-Transparent mode, placed into a high-impedance state. (Refer to Sections 6.2.4.8, 6.2.4.10, and Section 20.3 for further details.) <b>Note:</b> Refer to Section 3.2.1 for resistor requirements.
S_CLKRUN#	Secondary Clock Run (Transparent Mode)	1	I/O	L18	S_CLKRUN#: Valid only in Transparent mode. When driven high, slows down or stops the secondary PCI clock and is driven by a secondary PCI device to keep the clock running.
S_INTA#	Secondary Interrupt A# Output (Non-Transparent Mode)		OD		S_INTA#: In Non-Transparent mode, becomes S_INTA# output. Driven by the PCI 6540 to generate a PCI Interrupt request on the primary bus. In Universal Non-Transparent mode, placed into a high-impedance state. (Refer to Sections 6.2.4.8, 6.2.4.10, and Section 20.3 for further details.) <b>Note:</b> Refer to Section 3.2.1 for resistor requirements.
<b>Total</b>		3			

## 4 CLOCKING

This section describes the PCI 6540 clocking requirements.

To correctly operate, the PCI 6540 requires both a primary and secondary clock.

### 4.1 PRIMARY AND SECONDARY CLOCK INPUTS

The PCI 6540 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.

The PCI 6540 primary and secondary Clock inputs can be asynchronous. There are no skew constraints between the primary and secondary Clock inputs; however, the maximum ratio between the primary and secondary clock frequencies are 1:4 or 4:1.

The PCI 6540 operates at a maximum frequency of 133 MHz. The device provides five clock outputs (S\_CLKO[4:0]), which can be derived from P\_CLKIN, P\_CLKIN/2, or an external asynchronous clock source.

### 4.2 SECONDARY CLOCK OUTPUTS

**Note:** The secondary clock outputs provided by the PCI 6540 are not recommended for PCI-X use. Use high-quality clock buffers for PCI-X applications.

The PCI 6540 has five secondary clock outputs that can be used to drive up to four external secondary bus devices. S\_CLKO0 drives the PCI 6540 S\_CLKIN signal.

The rules for using secondary clocks are:

- Each secondary clock output is limited to no more than one PCI load
- Each clock trace length, including the feedback clock to the PCI 6540 S\_CLKIN signal, must have equal length and impedance
- Terminate or disable unused secondary clock outputs to reduce power dissipation and noise in the system

### 4.3 DISABLING SECONDARY CLOCK OUTPUTS

Secondary clock outputs may be disabled in two ways. If the S\_CLKOFF input is asserted (high), S\_CLKO0 is placed into a high-impedance state and S\_CLKO[4:1] are disabled and driven low.

The Clock Control register (Transparent mode—CLKCNTRL; PCI:68h, Non-Transparent mode—CLKCNTRL; PCI:94h) allows individual clock outputs to be disabled. Clock outputs disabled by CLKCNTRL remain disabled, regardless of S\_CLKOFF status.

### 4.4 USING AN EXTERNAL CLOCK SOURCE

S\_CLK\_STB input allows for an indication that the secondary external clock source is stable. If this input is de-asserted (low), then this indicates that the S\_CLKIN signal is not yet stable. The PCI 6540 does not de-assert S\_RSTOUT# until S\_CLK\_STB is asserted (high). This ensures that a valid and stable secondary clock source is present before transactions can occur on the secondary bus.

The PCI 6540 uses two signals, OSCSEL# and OSC\_IN, when connecting an external clock source to the PCI 6540. During normal operation, the PCI 6540 generates S\_CLKO[4:0] outputs, based on the PCI clock source (P\_CLKIN). If OSCSEL# is asserted (low), then the PCI 6540 derives S\_CLKO[4:0] from the OSCIN signal instead. Clock division is performed on the OSCIN and P\_CLKIN clocks, depending on the P\_M66EN and S\_M66EN signal states.

## 4.5 FREQUENCY DIVISION OPTIONS

The PCI 6540 has built-in frequency division options to automatically adjust the S\_CLKO[4:0] clocks for PCI 66 or 33 MHz operations. For PCI-X applications, use external, high-quality clock buffers and dividers. Table 4-1 lists the clock division ratios used, depending on P\_M66EN and S\_M66EN status.

**Note:** S\_M66EN cannot be floating.

**Table 4-1. PCI Clock Division Ratios**

P_M66EN	S_M66EN	Division Ratio
1	1	1/1
1	0	1/2
0	1	1/1
0	0	1/1

## 4.6 RUNNING SECONDARY PORT FASTER THAN PRIMARY PORT

The PCI 6540 allows the secondary port to use a higher clock frequency than that of the primary port. In this case, a secondary clock source, using an external oscillator or clock generator, must be provided.

If the external oscillator is connected to OSCIN and OSCSEL# is asserted (low), then the output generated by S\_CLKO[4:0] is divided, as per Table 4-1. Division control can be disabled by pulling S\_M66EN high and not connecting this pin to a PCI slot (which may be on the secondary bus). If the S\_CLKO[4:0] outputs are not required, then the external clock can be fed directly into the S\_CLKIN signal.

## 4.7 UNIVERSAL MODE CLOCK BEHAVIOR

The PCI 6540 clock behavior changes slightly when the device is configured in Universal mode.

In Universal Non-Transparent mode (U\_MODE=1 and TRANS#=1), input to the S\_CLKIN pin is ignored. Instead, the PCI 6540 uses the S\_CLKO0 pin as the secondary interface clock input. As a result, S\_CLKO0 operates as an input, rather than an output. The object of this is to allow the secondary interface clock to be derived from the PCI CLK of a CompactPCI backplane. When a suitably configured card is inserted in a CompactPCI system slot, the S\_CLKO0 output drives the CompactPCI CLK0 clock lines on the backplane. However, when the card is inserted in a peripheral slot, the S\_CLKO0 input is driven from the backplane clock.

If an external Clock buffer is used instead of S\_CLKO0 to drive the System Slot PCI Clock signal during PCI-X applications, then this output can be placed into a high-impedance state by setting the S\_CLKOFF input to 1.

## 4.8 PLL AND CLOCK JITTER

The PCI 6540 uses one PLL each for the primary and secondary interfaces. These PLLs can be individually disabled by connecting the P\_PPLEN# or S\_PPLEN# pin to 1.

The minimum input frequency of each PLL is 50 MHz. If a PCI 6540 port is used in a low-speed application (*for example*, at 33 MHz), then disable the appropriate PLL by setting P\_PPLEN# or S\_PPLEN# to high.

The PLL is sensitive to power and ground noise. To reduce power and ground bounce caused by the digital logic feeding into the PLLs, a dedicated set of PLL power and ground pins are provided. Connect the VDD pins for each PLL to a clean 2.5V supply and decouple to the appropriate ground pins.

Table 4-2 details the PLL operational parameters for both primary and secondary PLLs.

**Table 4-2. PLL and Clock Jitter Parameters**

Parameter	Minimum	Typical	Maximum	Unit	Condition
Input Frequency	50	—	133	MHz	—
Input Rise and Fall Time	—	—	500	ps	—
Input Cycle-to-Cycle Jitter	-100	—	+100	ps	—
Input Jitter Modulation Frequency	Must be < 100 KHz to allow PLL tracking or > 30 MHz to allow PLL filtering				
Output Cycle-to-Cycle Jitter	-150	—	+150	ps	Clean Power $V_{DD} = 2.5V$
Output Duty Cycle	45	—	55	%	Clean Power $V_{DD} = 2.5V$
Phase Lock Time	—	—	100	$\mu s$	Clean Power $V_{DD} = 2.5V$
PLL Power Dissipation	—	9	25	mW	Clean Power $V_{DD} = 2.5V$ $F_{in} = F_{out} = 133 \text{ MHz}$
Operating Temperature	-40	—	+125	$^{\circ}C$	—

## 4.9 DETECTING PCI BUS SPEED WITH THE REFERENCE CLOCK

The PCI 6540 has a reference clock input (REFCLK), which is used by a timer. Any fixed frequency source can be used as a reference clock source, although 14.318 MHz is recommended.

The timer can be set to time the primary or secondary port Clock inputs (P\_CLKIN or S\_CLKIN, respectively). The Timer Control register controls the count period and the PCI clock to be timed (TMRCNTRL[7:4 and 2:1]; PCI:61h, respectively). Software can then read the timer value from the Timer Counter register (TMRCNT; PCI:62h) and calculate the corresponding port's clock frequency.

## 4.10 PRIMARY OR SECONDARY CLOCK FREQUENCY MEASUREMENT

The Reference clock input signal, REFCLK, is used with the Timer Control and Timer Counter registers (TMRCNTRL; PCI:61h and TMRCNT; PCI:62h, respectively) to measure the approximate bus frequency on the primary or secondary interface.

The REFCLK clock frequency should be significantly slower than that of the primary and secondary clocks.

Software must select the target bus using bit 1 of the Timer Counter Clock Source Select bits (TMRCNTRL[2:1]; PCI:61h). Note that TMRCNTRL[2] is always cleared to 0. Software sets the Timer Enable bit to start the measurement (TMRCNTRL[0]=1; PCI:61h). TMRCNTRL[0] remains set, and software polls the Timer Stop bit until the bit is set to 1 (TMRCNTRL[3]=1; PCI:61h). When TMRCNTRL[3]=1, the measurement has finished and the result is presented in the Timer Counter register. To start a new measurement, the software must set TMRCNTRL[0] to 0, and then 1.

The measurement process counts the total measured bus clock rising edges that occurred during the overall counter period. The counter, however, accumulates only the bus clock rising edges that occurred during the high states of each Reference Clock cycle. The Count Period bit (TMRCNTRL[5:4]) values indicate the

number of high states used to accumulate the count (16, 32, 64, or 128). The total number of rising edges in all high states are accumulated and reported in the Timer Counter register. *For example*, if a reference clock speed of 14.318 MHz is used, the size of each Reference clock high state is  $(1/14.318\text{M})/2 = 349 \text{ ns}$ .

The Timer Counter register stores the accumulated count of rising edges within the Count Period. When the measurement is finished (indicated by TMRCNTRL[3]=1), the TMRCNT register value can be used to determine the approximate bus speed.

### Example:

- Speed of Reference Clock = 14.318 MHz
- Measured Clock = Secondary Bus Clock
- Count Period (Number of Windows; TMRCNTRL[5:4]=00b) = 16
- Secondary Bus Speed = 66 MHz

Based on the above configuration, TMRCNTRL[1] is set to 1 because the secondary bus clock is the measured clock. TMRCNTRL[5:4] are set to 00b for 16 high states. The software first writes 0 to TMRCNTRL[0] (assuming there is a previous measurement), then writes 1 to start the measurement. Software polls TMRCNTRL[3] until the bit is set to 1.

### Expected Timer Counter Count for this Example:

Because the reference clock is 14.318 MHz, each window size is about 349 ns.

If the measured clock speed is 66 MHz, the clock period is about 15 ns. Therefore, each window count is 23 or 24 ( $349 / 15$ ).

Because 16 windows (high states) were opened, the total count is in the range of  $23 \times 15 = 345$  and  $24 \times 15 = 360$ .

**Note:** For further details, refer to the TMRCNTRL and TMRCNT registers in Section 6, "Registers."



## 5 RESET AND INITIALIZATION

This section describes PCI-XCAP connections and operating frequency, secondary bus mode and frequency initialization, Conventional PCI mode 66 MHz operation, reset, and register initialization.

### 5.1 PCI-XCAP CONNECTIONS AND OPERATING FREQUENCY

The PCI 6540 operates at up to 66 MHz when working in Conventional PCI mode and up to 133 MHz in PCI-X mode. The primary and secondary ports each have a PCI-XCAP input (P\_XCAP and S\_XCAP\_IN, respectively), that determine whether that port is configured as a PCI or PCI-X port. In addition, PCI-XCAP is also used to determine the operating frequency of ports operating in PCI-X mode.

#### 5.1.1 Primary Port PCI-XCAP Connection

In a standard adapter card design, the P\_XCAP pin should be configured as detailed in *PCI-X r1.0b*, Section 6.2.

However, there may be situations where the standard PCI-X reset and initialization sequence is not available. *For example*, this may be the case during a CompactPCI Hot Swap insertion.

To force the primary port to use the PCI-X protocol, set the P\_XCAP input to 1. If P\_XCAP is set to 0, the primary port is configured to operate using the PCI protocol and standard PCI reset and initialization occurs.

#### 5.1.2 Secondary Port PCI-XCAP Connection

In Transparent mode, two PCI 6540 pins are associated with the secondary PCI-XCAP connection—S\_XCAP\_IN and S\_XCAP\_PU.

The S\_XCAP\_IN pin is used to determine whether secondary devices are capable of PCI-X operation and at what frequency the devices can operate.

Connect S\_XCAP\_IN to a weak 56K-Ohm resistor, pulled up to 3.3V, and to the PCI-XCAP pin on secondary PCI-X slots. Connect S\_XCAP\_PU output and S\_XCAP\_IN together with a 1K-Ohm resistor.

During PCI-X and frequency discovery on the secondary bus, the S\_XCAP\_PU signal is driven high and provides a strong pull-up resistor. This allows capacitors attached to PCI-X adapter card PCI-XCAP pins to charge. The PCI 6540 can then determine whether PCI-X cards are attached to the secondary port.

If PCI-X and frequency discovery are not required, set S\_XCAP\_IN to 1 to force PCI-X transactions to occur on the secondary bus, or to 0 to force Conventional PCI transactions.

In Non-Transparent mode, S\_XCAP\_PU output is always floating and S\_XCAP\_IN functions as described in Section 5.1.1.

### 5.2 SECONDARY BUS MODE AND FREQUENCY INITIALIZATION SEQUENCE

The PCI 6540 places its secondary bus in PCI-X mode based on the capabilities of devices connected to the secondary bus, independent of the primary bus mode. If only one side of a bridge is operating in PCI-X mode, the PCI 6540 translates the protocol between the two buses.

At the rising edge of P\_RSTIN#, the PCI 6540 latches the frequency and mode of its primary bus. For Transparent mode, the PCI 6540 initializes the secondary bus as follows:

1. Senses the state of S\_XCAP\_IN and S\_M66EN for all devices on the secondary bus and selects the appropriate mode and clock frequency. When:
  - Secondary bus includes one or more 33 MHz Conventional PCI devices, the bus operating frequency must be Conventional PCI 33 MHz.
  - Only Conventional PCI 66 MHz devices are present on the secondary bus, the operating frequency is 66 MHz.
  - Secondary bus includes only PCI-X 133 devices, the maximum clock frequency is 133 MHz.
  - Secondary bus includes only PCI-X 100 devices, the maximum clock frequency is 100 MHz. 100 MHz selection status can be configured using the GPIO13 pin.

- Secondary bus includes only PCI-X 66 devices, the maximum clock frequency is 66 MHz.
2. Asserts the appropriate signals for the PCI-X initialization pattern on the secondary bus.
  3. De-asserts S\_RSTOUT# to place all devices on the secondary bus in the appropriate mode.

### 5.3 CONVENTIONAL PCI MODE 66 MHz OPERATION

The P\_M66EN and S\_M66EN signals indicate whether the primary and secondary interfaces are operating at 66 MHz. This information is needed to control the secondary bus frequency. *PCI r2.3* prohibits PCI clock frequency changes above 33 MHz, except during a PCI reset.

The following primary and secondary bus frequency combinations are supported when using the primary P\_CLKIN signal to generate the secondary clock outputs:

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

If P\_M66EN is low (*for example*, the primary bus runs at 33 MHz), the PCI 6540 drives S\_M66EN low to indicate that the secondary bus is operating at 33 MHz. If the secondary bus is set to run faster than the primary bus, S\_M66EN need not be connected to secondary PCI devices.

The PCI 6540 can also generate S\_CLKO[4:0] outputs from OSCIN, if enabled. When the PCI 6540 is using asynchronous clock inputs (*for example*, S\_CLKO[4:0] are not derived from P\_CLKIN or OSCIN), the previously listed frequencies are not the only possible clock combinations. If an external clock is used for the secondary interface, the PCI 6540 operates with a maximum ratio of 1:4 or 4:1 between the primary and secondary bus clocks.

### 5.4 RESET

This subsection describes the primary and secondary interface, and chip reset mechanisms. The PCI 6540 has three reset inputs—PWRGD, P\_RSTIN#, and S\_RSTIN#. In addition, the PCI 6540 can respond to Power Management-initiated internal reset and software-controlled internal reset.

After the reset signals are de-asserted, the PCI 6540 requires 512 clocks to initialize bridge functions. During this initialization, Type 0 accesses can be accepted. If cross-bridge traffic is presented to the PCI 6540, the device changes to an unknown state.

**Note:** Care must be taken when using P\_RSTOUT# and S\_RSTOUT# to feed to their corresponding reset input signals (P\_RSTIN# and S\_RSTIN#, respectively). P\_RSTIN# can cause S\_RSTOUT# assertion and S\_RSTIN# can cause P\_RSTOUT# assertion. Therefore, there is the potential to lock PCI 6540 in a permanent Reset cycle if both Reset outputs are fed back to their corresponding inputs.

#### 5.4.1 Power Good Reset

When PWRGD is not active, the following occurs:

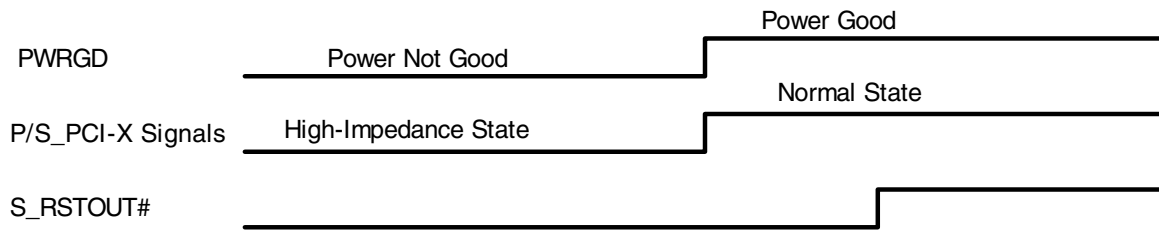
1. PCI 6540 immediately places all primary PCI-X interface signals (*except* P\_RSTOUT#) into a high-impedance state.
2. PCI 6540 performs a full chip reset.
3. P\_RSTOUT# is not asserted in Transparent mode, but is asserted in Non-Transparent mode. S\_RSTOUT# is asserted (low).
4. All registers and extended registers with default values are reset.
5. If P\_RSTIN# is low, PWRGD going from low-to-high causes the serial EEPROM to be loaded.

**Note:** The PCI 6540 requires a clean low-to-high transition for PWRGD input. The PWRGD signal is not internally debounced—it must be externally debounced and a high input must reflect that the power is stable (Non-Transparent mode only).

The asserting and de-asserting edges of PWRGD can be asynchronous to P\_CLKIN and S\_CLKIN. Usually, PWRGD should not change to low when P\_RSTIN# and/or S\_RSTIN# are high.

If P\_RSTIN# is de-asserted before PWRGD assertion, the primary PCI-X signals remain in a high-impedance state because PWRGD is not asserted. S\_RSTOUT# remains asserted until a few clocks after PWRGD assertion. Similarly, if S\_RSTIN# is de-asserted before PWRGD is asserted, the secondary PCI-X signals remain in a high-impedance state.

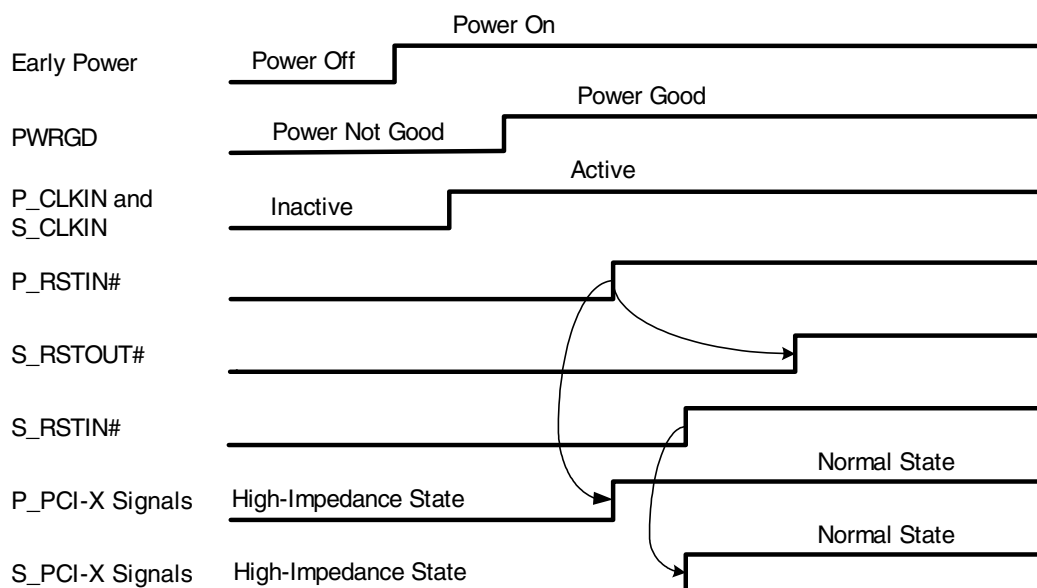
When PWRGD is de-asserted, all primary and secondary PCI signals are placed into a high-impedance state. S\_RSTOUT# assertion is extended until PWRGD is asserted. (Refer to Timing Diagram 5-1.)



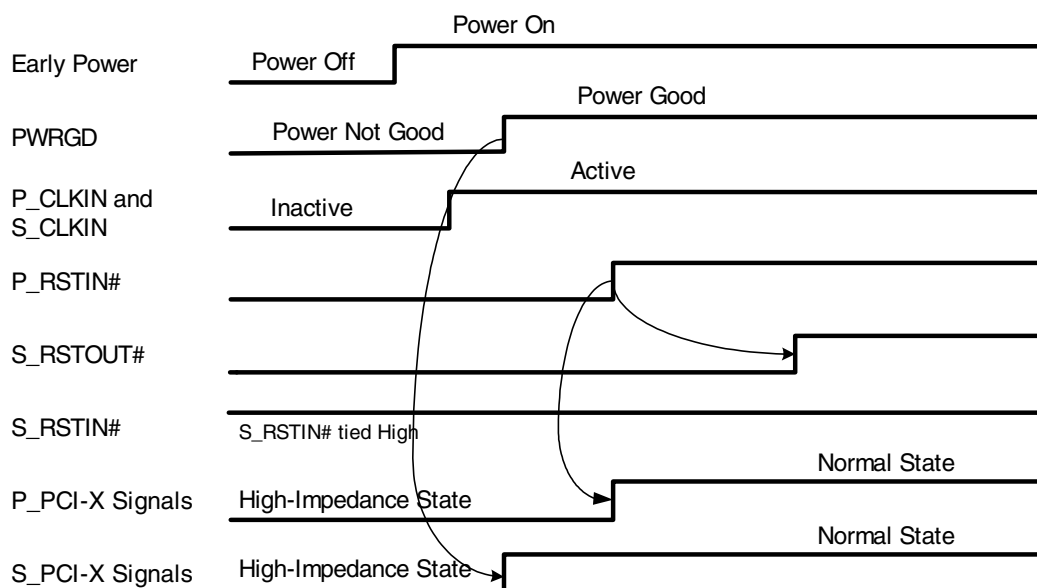
Timing Diagram 5-1. PWRGD Assertion

### 5.4.1.1 PWRGD, Resets, and PCI-X Signals Relationship

The following diagrams illustrate the timing relationship between the PWRGD, P\_RSTIN#, S\_RSTIN# and PCI-X signals. In general, ensure PWRGD is ahead of P\_RSTIN# and S\_RSTIN#.

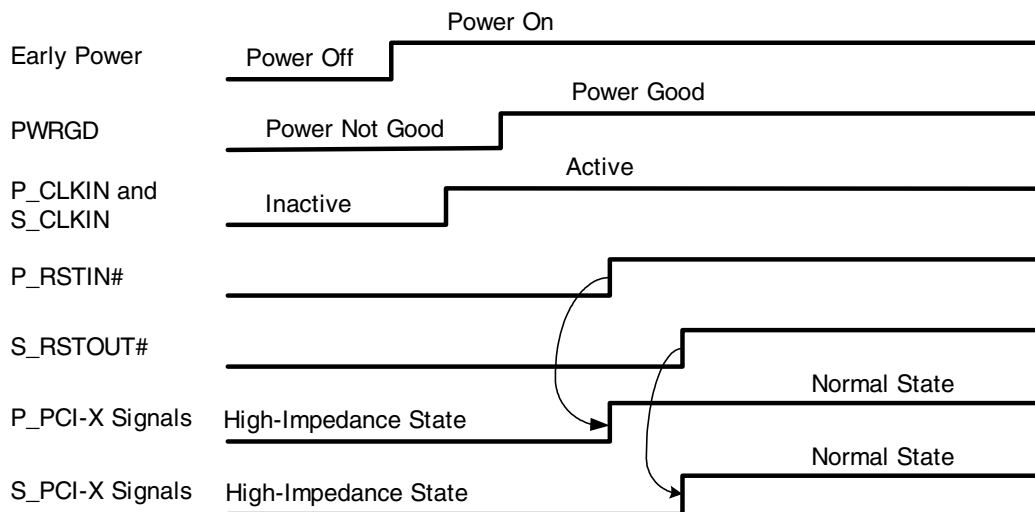


Timing Diagram 5-2. PWRGD and Resets where S\_RSTIN# Is De-Asserted Late—Universal Transparent Mode



Timing Diagram 5-3. PWRGD and Resets where S\_RSTIN# Is Tied High—Universal Transparent Mode

In Timing Diagram 5-4, the PCI 6540 is using Universal Non-Transparent mode. Therefore, the S\_RSTOUT# pin acts as the secondary port reset input (equivalent to S\_RSTIN#) and S\_RSTIN# is not used.



Timing Diagram 5-4. PWRGD and Resets—Universal Non-Transparent Mode

### 5.4.2 Primary Reset Input

When P\_RSTIN# is asserted, the following events occur:

1. The PCI 6540 immediately places all primary PCI interface signals (*except* P\_RSTOUT#) into a high-impedance state.
2. Transparent mode:
  - All registers (*except* Sticky Scratch registers, SCRATCHx; EXT:00h to 07h) are reset.
  - S\_RSTOUT# is driven active to indicate a primary PCI reset.
- Non-Transparent mode:
  - Primary Configuration registers at offsets 00h to 3Fh and Shadow registers at offsets 40h to 77h are reset.
  - Upon P\_RSTIN# de-assertion, S\_INTA# is asserted and the appropriate Status bit set (DWNINTSR[4]=1; PCI:CAh) if the primary PCI Interrupt Event is enabled (DWNINTE[4]=1; PCI:CFh).
  - PCI 6540 performs a Primary Port reset. The secondary port and Sticky Scratch registers are not affected.
3. In Transparent and Universal Transparent modes, P\_RSTIN# assertion automatically causes a secondary port reset and S\_RSTOUT# is asserted. S\_TRDY#, S\_DEVSEL#, and S\_STOP# are driven for PCI-X speed inquiry.
4. Clock Disable bits (Transparent mode—CLKCNTRL[8:0]; PCI:68h, Non-Transparent mode—CLKCNTRL[8:0]; PCI:94h) are shifted in at P\_RSTIN# de-assertion.

The asserting and de-asserting edges of P\_RSTIN# can be asynchronous to P\_CLKIN and S\_CLKIN.

When P\_RSTIN# is asserted, all primary PCI interface signals, including the primary Request output, are immediately placed into a high-impedance state. All Posted Write and Delayed Transaction data buffers are reset. Therefore, transactions residing in the buffers are discarded upon P\_RSTIN# assertion.

### 5.4.3 Primary Reset Output

When operating in Non-Transparent mode, the PCI 6540 asserts P\_RSTOUT# when one of the following conditions is met:

- PWRGD input is asserted low in Non-Transparent mode (TRANS#=1).
- S\_RSTIN# (using the S\_RSTOUT# pin) is asserted in Universal Non-Transparent mode (U\_MODE=1 and TRANS#=1).
- S\_RSTIN# is asserted in standard Non-Transparent mode, with the primary port having boot priority (TRANS#=1 and P\_BOOT=1).
- Non-Transparent Diagnostic Control register Primary Reset bit is set (DCNTRL[5]=1; PCI:D9h).  
P\_RSTOUT# remains asserted until the Primary Reset Output Mask bit is cleared (DCNTRL[4]=0; PCI:D9h).

P\_RSTOUT# is used only for Non-Transparent mode operation, and is not valid when the PCI 6540 is in Transparent mode.

### 5.4.4 Secondary Reset Input

The S\_RSTIN# pin is used only during Non-Transparent mode. In Universal Non-Transparent mode, the S\_RSTOUT# pin functions as the secondary port Reset Input pin. When S\_RSTIN# is asserted, the following events occur:

1. PCI 6540 immediately places all secondary PCI interface signals (*except* S\_RSTOUT#) into a high-impedance state.
2. Non-Transparent mode secondary Configuration registers at offsets 00h to 3Fh are reset.
3. P\_RSTOUT# is driven active to indicate a secondary PCI reset if the Primary Reset Output Mask bit is not set (DCNTRL[4]=0; PCI:D9h).
4. Upon S\_RSTIN# de-assertion, P\_INTA# is driven active and the Status bit is set if the secondary PCI Interrupt Event is enabled (UPSINTSR[4]=1; PCI:CEh and UPSINTE[4]=1; PCI:CBh, respectively).
5. PCI 6540 performs a Secondary Port reset. The primary port and Sticky Scratch registers (SCRATCHx; EXT:00h to 07h) are not affected.

The asserting and de-asserting edges of S\_RSTIN# can be asynchronous to P\_CLKIN and S\_CLKIN.

**Note:** When not used, connect or pull S\_RSTIN# high.

In Non-Transparent mode, when S\_RSTIN# is asserted, all secondary PCI interface signals, including the secondary grant outputs, are immediately placed into a high-impedance state. All Posted Write and Delayed Transaction data buffers are reset. Therefore, transactions residing in the buffers are discarded upon S\_RSTIN# assertion.

When S\_RSTOUT# is asserted by way of the Secondary Reset bit (Transparent mode—BCNTRL[6]=1; PCI:3Eh, Non-Transparent mode—BCNTRL[6]=1; PCI:42h Shadow register), the PCI 6540 remains accessible during secondary reset and continues to respond to accesses to its Configuration space from the primary interface.

#### 5.4.4.1 Universal Mode Secondary Reset Input

In Universal Non-Transparent mode (U\_MODE=1 and TRANS#=1), S\_RSTOUT# is disabled and the S\_RSTOUT# pin is used as the equivalent of the S\_RSTIN# pin. During this mode, a low input presented at S\_RSTOUT# causes a Secondary Port reset. S\_RSTIN# is not used in this mode.

#### 5.4.5 Secondary Reset Output

The PCI 6540 asserts S\_RSTOUT# when one of the following conditions is met:

- P\_RSTIN# is asserted. S\_RSTOUT# remains asserted when one of the following conditions is met:
  - P\_RSTIN# is asserted Secondary Clock Serial Disable Mask (Transparent mode—CLKCNTRL[8:0]; PCI:68h, Non-Transparent mode—CLKCNTRL[8:0]; PCI:94h) is being shifted in (16 Clock cycles after P\_RSTIN# de-assertion), using MSK\_IN and GPIO[2, 0]
  - S\_CLKIN\_STB is low
- Diagnostic Control register Chip Reset and Bridge Control Secondary Reset bits are set (Transparent mode—DCNTRL[0]=1; PCI:41h and BCNTRL[6]=1; PCI:3Eh, Non-Transparent mode—DCNTRL[0]=1; PCI:D9h and BCNTRL[6]=1; PCI:42h, respectively)

S\_RSTOUT# remains asserted until the Secondary Reset Output Mask bit is cleared (Transparent mode—DCNTRL[3]=0; PCI:41h, Non-Transparent mode—DCNTRL[3]=0; PCI:D9h).

In Transparent mode, or Non-Transparent mode with P\_BOOT=0, when there is a D<sub>3hot</sub>-to-D<sub>0</sub> transition with the Power Management Control/Status register Power State bits programmed to D<sub>0</sub> (PMCSR[1:0]=00b; PCI:E0h), S\_RSTOUT# is active for 16 primary port PCI Clock cycles.

In Transparent and Universal Transparent modes, while S\_RSTOUT# is asserted, S\_DEVSEL#, S\_STOP#, and S\_TRDY# are driven for PCI-X speed inquiry.

#### 5.4.6 Software Resets

The Diagnostic Control register Chip Reset bit can be used to reset the PCI 6540 as the PWRGD input (Transparent mode—DCNTRL[0]=1; PCI:41h, Non-Transparent mode—DCNTRL[0]=1; PCI:D9h). This action does not cause P\_RSTOUT# assertion. However, the action does cause S\_RSTOUT# assertion and the signals are not placed into a high-impedance state. Additionally, if the PCI 6540 is in Non-Transparent mode, the serial EEPROM autoloads.

When the Chip Reset bit is set, all registers and chip states are reset. When chip reset completes, within four PCI Clock cycles after completion of the Configuration Write operation that sets the Chip Reset bit, the Chip Reset bit automatically clears and the PCI 6540 is ready for configuration. During chip reset, the PCI 6540 is inaccessible.

#### 5.4.7 Power Management Internal Reset

In Transparent mode, or Non-Transparent mode with P\_BOOT=0, when there is a D<sub>3hot</sub>-to-D<sub>0</sub> transition with the Power Management Control/Status register Power State bits programmed to D<sub>0</sub> (PMCSR[1:0]=00b; PCI:E0h), an internal reset equivalent to P\_RSTIN# is generated and all relevant registers are reset. However, P\_RSTOUT# and S\_RSTOUT# are **not** asserted.

In Non-Transparent mode with P\_BOOT=1, when there is a D<sub>3hot</sub>-to-D<sub>0</sub> transition with the Power State bits programmed to D<sub>0</sub>, P\_RSTOUT# is **not** asserted.

## 5.4.8 Reset Inputs Table

There are other Transparent and Non-Transparent mode-related reset controls that can be used to generate primary or secondary Reset outputs. Table 5-1 depicts the effect of various Reset inputs on the PCI 6540.

Table 5-1. Reset Input Effect on PCI 6540

Operating Mode Reset Inputs	Transparent Mode	Non-Transparent Mode	Universal Transparent Mode	Universal Non-Transparent Mode (S_RSTOUT# Used as Secondary Reset Input)
P_RSTIN#	<ul style="list-style-type: none"> <li>Resets primary and secondary ports</li> <li>Asserts S_RSTOUT#</li> <li>Causes serial EEPROM load</li> </ul>	<ul style="list-style-type: none"> <li>Resets only primary port</li> <li>Asserts S_RSTOUT#</li> <li>Causes serial EEPROM load</li> </ul>	<ul style="list-style-type: none"> <li>Resets primary and secondary ports</li> <li>Asserts S_RSTOUT#</li> <li>Causes serial EEPROM load</li> </ul>	<ul style="list-style-type: none"> <li>Resets only primary port</li> <li>Causes serial EEPROM load</li> </ul>
S_RSTIN#	<i>Not Used</i>	<ul style="list-style-type: none"> <li>Resets only secondary port</li> <li>Asserts P_RSTOUT#</li> </ul>	<i>Not Used</i>	<i>Not Used</i>
S_RSTOUT#	Not used as input	Not used as input	Not used as input	<ul style="list-style-type: none"> <li>Used as reset input and resets only secondary port</li> <li>Asserts P_RSTOUT#</li> </ul>
S_CLK_STB not active	<ul style="list-style-type: none"> <li>Resets only secondary port</li> <li>Asserts S_RSTOUT#</li> </ul>	Asserts S_RSTOUT#	<ul style="list-style-type: none"> <li>Resets only secondary port</li> <li>Asserts S_RSTOUT#</li> </ul>	No effect
PWRGD not ready	<ul style="list-style-type: none"> <li>Asserts S_RSTOUT#</li> <li>Resets Sticky Scratch registers (SCRATCHx; EXT:00h – 07h)</li> <li>Causes serial EEPROM load</li> </ul>	<ul style="list-style-type: none"> <li>Asserts P_RSTOUT# and S_RSTOUT#</li> <li>Causes serial EEPROM load</li> <li>Resets Sticky Scratch registers</li> </ul>	<ul style="list-style-type: none"> <li>Asserts S_RSTOUT#</li> <li>Resets Sticky Scratch registers</li> <li>Causes serial EEPROM load</li> </ul>	<ul style="list-style-type: none"> <li>Asserts P_RSTOUT#</li> <li>Causes serial EEPROM load</li> <li>Resets Sticky Scratch registers</li> </ul>
Transparent mode Chip Reset (DCNTRL[0]=1; PCI:41h)	Resets internal state machines and S_RSTIN#	N/A	Resets internal state machines and S_RSTIN#	N/A
Non-Transparent mode Chip Reset (DCNTRL[0]=1; PCI:D9h)	N/A	<ul style="list-style-type: none"> <li>Resets internal state machines</li> <li>Causes serial EEPROM load</li> </ul>	N/A	Resets internal state machines
Transparent mode Secondary Reset (BCNTRL[6]=1; PCI:3Eh)	Asserts S_RSTOUT#	N/A	Asserts S_RSTOUT#	N/A
Non-Transparent mode Secondary Reset (BCNTRL[6]=1; PCI:42h Shadow register)	N/A	Causes S_RSTOUT# active	N/A	No effect
Non-Transparent mode Primary Reset (DCNTRL[5]=1; PCI:D9h)	N/A	Causes P_RSTOUT# to be active	N/A	Causes P_RSTOUT# to be active



### 5.4.9 Power-Up and Reset Pin State Table

The PCI 6540 supports PWRGD, P\_RSTIN#, S\_RSTIN#, and device hiding. Table 5-2 depicts the pin state for each event. With the exception of the

Power Not Good State (PWRGD=0), all states assume a valid input clock.

For device hiding, the pin follows the corresponding mode's value to the left, except for differences listed in the "Device Hiding Ejector Switch Open, P\_RSTIN#=1 and S\_RSTIN#=1" column.

**Table 5-2. Pin State during PWRGD, P\_RSTIN#, S\_RSTIN#, and Device Hiding**

Power-Up/Reset PCI 6540 Pins	PWRGD=0, with or without Clock, P_RSTIN#=X	Transparent Mode, PWRGD=1 and P_RSTIN#=0	Non-Transparent Mode, P_RSTIN#=0 and S_RSTIN#=0	Non-Universal, Non-Transparent Mode, P_RSTIN#=1 and S_RSTIN#=0	Universal Non-Transparent Mode, P_RSTIN#=1 and S_RSTIN#=0	Device Hiding Ejector Switch Open, P_RSTIN#=1 and S_RSTIN#=1
BPCC_EN	I	I	I	I	I	
DEV64#	I	I	I	I	I	
EEPCLK	No P_CLKIN: U With P_CLKIN: D1	D1	D1	D1	D1	
EEPDATA	No P_CLKIN: U With P_CLKIN: D1	D1	D1	D1	D1	
EJECT	I	I	I	I	I	
ENUM#	T	T	T	T	T	
GPIO0	D1	D1	D0	D01	D01	T if not used as output
GPIO[2:1]	D0	D0	D0	D01	D01	T if not used as output
GPIO[15:3}	T	T	T	T	T	
L_STAT - EJECT=0	D1	D1	D1	D01	D01	D0
MSK_IN	I	I	I	I	I	
OSCIN	I	I	I	I	I	
OSCSEL#	I	I	I	I	I	
P_ACK64#	T	T	T	T	T	
P_AD[63:0]	T	T	T	T	T	
P_AV <sub>DD</sub>	I	I	I	I	I	
P_AV <sub>SS</sub>	I	I	I	I	I	

Table 5-2. Pin State during PWRGD, P\_RSTIN#, S\_RSTIN#, and Device Hiding (Continued)

Power-Up/Reset PCI 6540 Pins	PWRGD=0, with or without Clock, P_RSTIN#=X	Transparent Mode, PWRGD=1 and P_RSTIN#=0	Non-Transparent Mode, P_RSTIN#=0 and S_RSTIN#=0	Non-Universal, Non-Transparent Mode, P_RSTIN#=1 and S_RSTIN#=0	Universal Non-Transparent Mode, P_RSTIN#=1 and S_RSTIN#=0	Device Hiding Ejector Switch Open, P_RSTIN#=1 and S_RSTIN#=1
P_BOOT	I	I	I	I	I	
P_CBE[7:0]#	T	T	T	T	T	
P_CLKIN	I	I	I	I	I	
P_CLKOE	I	I	I	I	I	
P_CLKRUN#	I	I	I	I	I	
P_CR	I	I	I	I	I	
P_DEVSEL#	T	T	T	T	T	
P_FRAME#	T	T	T	T	T	
P_GNT#	I	I	I	I	I	
P_IDSEL	I	I	I	I	I	
P_INTA# (Non-Transparent Mode)	T	T	T	T	T	
P_IRDY#	T	T	T	T	T	
P_LOCK#	T	T	T	T	T	
P_M66EN#	I	I	I	I	I	
P_PAR	T	T	T	T	T	
P_PAR64	T	T	T	T	T	
P_PERR#	T	T	T	T	T	
P_PLEN#	I	I	I	I	I	
P_PME#	T	T	T	T	T	
P_REQ#	T	T	T	D1	D1	
P_REQ64#	T	T	T	T	T	
P_RSTIN#	I	I	I	I	I	

Table 5-2. Pin State during PWRGD, P\_RSTIN#, S\_RSTIN#, and Device Hiding (Continued)

Power-Up/Reset PCI 6540 Pins	PWRGD=0, with or without Clock, P_RSTIN#=X	Transparent Mode, PWRGD=1 and P_RSTIN#=0	Non-Transparent Mode, P_RSTIN#=0 and S_RSTIN#=0	Non-Universal, Non-Transparent Mode, P_RSTIN#=1 and S_RSTIN#=0	Universal Non-Transparent Mode, P_RSTIN#=1 and S_RSTIN#=0	Device Hiding Ejector Switch Open, P_RSTIN#=1 and S_RSTIN#=1
P_RSTOUT#	D1 (Transparent mode) D0 (Non-Transparent mode)	D1	D0	D0	D0	D1
PRV_DEV	I	I	I	I	I	
P_SERR#	T	T	T	T	T	
P_STOP#	T	T	T	T	T	
P_TRDY#	T	T	T	T	T	
P_TST[1:0]	I	I	I	I	I	
PWRGD	I	I	I	I	I	
P_XCAP	I	I	I	I	I	
REFCLK	I	I	I	I	I	
S_ACK64#	T	T	T	T	T	
S_AD[63:0]	T	T	T	T	T	
S_AV <sub>DD</sub>	I	I	I	I	I	
S_AV <sub>SS</sub>	I	I	I	I	I	
S_CBE[7:0]#	T	T	T	T	T	
S_CFN#	I	I	I	I	I	
S_CLKIN	I	I	I	T	I – <i>Not Used</i>	
S_CLKIN_STB	I	I	I	I	I	
S_CLKO0 - S_CLKOFF=0 - S_CLKOFF=1	D01 D0	D01 D0	D01 D0	D01 D0	I I	
S_CLKO[4:1] - S_CLKOFF=0 - S_CLKOFF=1	D01 D0	D01 D0	D01 D0	D01 D0	D01 D0	
S_CLKOFF	I	I	I	I	I	

Table 5-2. Pin State during PWRGD, P\_RSTIN#, S\_RSTIN#, and Device Hiding (Continued)

Power-Up/Reset PCI 6540 Pins	PWRGD=0, with or without Clock, P_RSTIN#=X	Transparent Mode, PWRGD=1 and P_RSTIN#=0	Non-Transparent Mode, P_RSTIN#=0 and S_RSTIN#=0	Non-Universal, Non-Transparent Mode, P_RSTIN#=1 and S_RSTIN#=0	Universal Non-Transparent Mode, P_RSTIN#=1 and S_RSTIN#=0	Device Hiding Ejector Switch Open, P_RSTIN#=1 and S_RSTIN#=1
S_CR	I	I	I	I	I	
S_DEVSEL#	T	T (PCI) D01 (PCI-X)	T	T	T	
S_FRAME#	T	T	T	T	T	
S_GNT0#	T	D1	T	T	I	
S_GNT[7:1]#	T	D1	T	T	T	
S_IDSEL	I	I	I	I	I	
S_INTA# or S_CLKRUN# if in Non-Universal Transparent mode	T D0 if Non-Universal Transparent mode	D0	T	T	T	
S_IRDY#	T	T	T	T	T	
S_LOCK#	T	T	T	T	T	
S_M66EN#	I	I	I	I	I	
S_PAR	T	T	T	T	T	
S_PAR64	T	T	T	T	T	
S_PERR#	T	T	T	T	T	
S_PLEN#	I	I	I	I	I	
S_PME#	T	T	T	T	T	
S_REQ0#	T	I	I	I	D1	
S_REQ[7:1]#	I	I	I	I	I	
S_REQ64#	T	D0	T	T	T	
S_RSTIN#	I	I	I	I	I	
S_RSTOUT#	D0	D0	D0	D01	I	D1 I in Universal Non-Transparent mode

Table 5-2. Pin State during PWRGD, P\_RSTIN#, S\_RSTIN#, and Device Hiding (Continued)

Power-Up/Reset PCI 6540 Pins	PWRGD=0, with or without Clock, P_RSTIN#=X	Transparent Mode, PWRGD=1 and P_RSTIN#=0	Non-Transparent Mode, P_RSTIN#=0 and S_RSTIN#=0	Non-Universal, Non-Transparent Mode, P_RSTIN#=1 and S_RSTIN#=0	Universal Non-Transparent Mode, P_RSTIN#=1 and S_RSTIN#=0	Device Hiding Ejector Switch Open, P_RSTIN#=1 and S_RSTIN#=1
S_SERR#	T	T	T	T	T	
S_STOP#	T	T (PCI) D01 (PCI-X)	T	T	T	
S_TRDY#	T	T (PCI) D01 (PCI-X)	T	T	T	
S_TST[1:0]	I	I	I	I	I	
S_XCAP_IN	I	I	I	I	I	
S_XCAP_PU	I	I	I	I	I	
TEST	I	I	I	I	I	
V <sub>DD_CORE</sub>	I	I	I	I	I	
V <sub>DD_IO</sub>	I	I	I	I	I	
V <sub>SS</sub>	I	I	I	I	I	
XB_MEM (Non-Transparent Mode)	I	I	I	I	I	

**Legend:***U* = Undetermined*I* = Input Only*T* = Placed into a high-impedance state*D1* = Drive 1 to output*D0* = Drive 0 to output*D01* = Can drive both 0 or 1 to output

## 5.5 REGISTER INITIALIZATION

The PCI 6540 Configuration registers may be initialized in one of three ways:

- Default values
- Serial EEPROM contents
- Host initialization

### 5.5.1 Default Initialization

After P\_RSTIN# de-assertion or PWRGD assertion (whichever occurs later), the PCI 6540 automatically checks for a valid a serial EEPROM. If the serial EEPROM is not valid nor present, the PCI 6540 automatically loads the default values into the Configuration registers. (Refer to the “Value after Reset” column of the tables in Section 6, “Registers.”)

The Sticky Scratch registers (SCRATCHx; EXT:00h to 07h) are cleared only if PWRGD is de-asserted.

### 5.5.2 Serial EEPROM Initialization

After P\_RSTIN# de-assertion or PWRGD assertion (whichever occurs later), if the PCI 6540 finds a valid serial EEPROM, register values are loaded from the serial EEPROM and overwrite the default values.

### 5.5.3 Host Initialization

When device initialization is complete, the host system may access the appropriate registers to configure them according to system requirements.

Typically, registers are accessed by performing Type 0 Configuration accesses from the appropriate bus. The exceptions to this are the Extended registers, which are accessed using the Extended Register Index (EXTRIDX; PCI:D3h) and the Extended Register Data (EXTRDATA; PCI:D4h).

For details regarding register access, refer to Section 6, “Registers.”

**Note:** *Not all registers may be written to or may not be available from both sides of the bridge.*

## 6 REGISTERS

This section describes the PCI 6540 Transparent and Non-Transparent mode PCI and PCI-X registers.

As a Transparent PCI-X bridge, the PCI 6540 includes the standard Type 01h Configuration Space header as defined in *P-to-P Bridge r1.1*. In Conventional PCI mode, these registers operate as defined in this specification. If either PCI 6540 interface is initialized to PCI-X mode, the requirements for these registers change to meet those of *PCI-X r1.0b, Section 8.6*.

As a Non-Transparent PCI-X bridge, the PCI 6540 includes the standard Type 0h Configuration Space header as defined in *PCI r2.3*. In Conventional PCI mode, these registers operate as defined in this specification. If either PCI 6540 interface is initialized to PCI-X mode, the requirements for these registers change to meet those of *PCI-X r1.0b, Section 7*.

**Note:** Registers listed with a PCI offset or address are accessed by standard PCI Type 0 Configuration accesses.

## 6.1 PCI CONFIGURATION REGISTER ADDRESS MAPPING—TRANSPARENT MODE

Table 6-1. PCI Configuration Register Address Mapping—Transparent Mode

PCI Configuration Register Address	To ensure software compatibility with other versions of the PCI 6540 family and to ensure compatibility with future enhancements, write 0 to all unused bits.								PCI Writable	Serial EEPROM Writable
	31	24	23	16	15	8	7	0		
00h	Device ID*				Vendor ID*				Yes	Yes
04h	Primary Status				Primary Command				Yes	No
08h	Class Code*						Revision ID		Yes	Yes
0Ch	Built-In Self-Test ( <i>Not Supported</i> )		Header Type*		Primary Latency Timer		Cache Line Size		Yes	Yes
10h – 14h	Reserved								No	No
18h	Secondary Latency Timer		Subordinate Bus Number		Secondary Bus Number		Primary Bus Number		Yes	No
1Ch	Secondary Status				I/O Limit		I/O Base		Yes	No
20h	Memory Limit				Memory Base				Yes	No
24h	Prefetchable Memory Limit				Prefetchable Memory Base				Yes	No
28h	Prefetchable Memory Base Upper 32 Bits								Yes	No
2Ch	Prefetchable Memory Limit Upper 32 Bits								Yes	No
30h	I/O Limit Upper 16 Bits				I/O Base Upper 16 Bits				Yes	No
34h	Reserved						New Capability Pointer		No	No
38h	Reserved								No	No
3Ch	Bridge Control				Interrupt Pin		Reserved		Yes	No
40h	Arbiter Control				Diagnostic Control		Chip Control		Yes	No
44h	Miscellaneous Options				Timeout Control		Primary Flow-Through Control		Yes	Yes
48h	Secondary Incremental Prefetch Count		Primary Incremental Prefetch Count		Secondary Initial Prefetch Count		Primary Initial Prefetch Count		Yes	Yes
4Ch	Buffer Control		Secondary Flow-Through Control		Secondary Maximum Prefetch Count		Primary Maximum Prefetch Count		Yes	Yes
50h	Reserved		Test		Internal Arbiter Control				Yes	No
54h	Serial EEPROM Data				Serial EEPROM Address		Serial EEPROM Control		Yes	No
58h	Reserved								No	No
5Ch	Reserved								No	No
60h	Timer Counter				Timer Control		Reserved		Yes	No
64h	GPIO[3:0] Input Data		GPIO[3:0] Output Enable		GPIO[3:0] Output Data		P_SERR# Event Disable		Yes	No
68h	Clock Run		P_SERR# Status		Clock Control				Yes	No
6Ch	Private Memory Limit				Private Memory Base				Yes	No
70h	Private Memory Base Upper 32 Bits								Yes	No
74h	Private Memory Limit Upper 32 Bits								Yes	No
78h – 98h	Reserved								No	No



Table 6-1. PCI Configuration Register Address Mapping—Transparent Mode (Continued)

PCI Configuration Register Address	To ensure software compatibility with other versions of the PCI 6540 family and to ensure compatibility with future enhancements, write 0 to all unused bits.							PCI Writable	Serial EEPROM Writable	
	31	24	23	16	15	8	7			0
9Ch	GPIO[7:4] Input Data		GPIO[7:4] Output Enable		GPIO[7:4] Output Data		Hot Swap Switch and ROR Control		Yes	No
A0h	GPIO[15:8] Input Data		GPIO[15:8] Output Enable		GPIO[15:8] Output Data		Power-Up Status		Yes	No
ACh – CCh	Reserved								No	No
D0h	Extended Register Index		Reserved						Yes	No
D4h	Extended Register Data								Yes	No
D8h	Reserved								No	No
DCh	Power Management Capabilities*				Power Management Next Capability Pointer (E4h)		Power Management Capability ID (01h)		Yes	Yes
E0h	Power Management Data*		PMCSR Bridge Supports Extensions		Power Management Control/Status*			Yes	Yes	
E4h	Reserved		Hot Swap Control/Status (0h)		Hot Swap Next Capability Pointer (E8h)		Hot Swap Control (Capability ID) (06h)		Yes	No
E8h	VPD Address (0h)				VPD Next Capability Pointer (F0h)		VPD Capability ID (03h)		Yes	No
ECh	VPD Data (0h)								Yes	No
F0h	PCI-X Secondary Status				PCI-X Next Capability Pointer (0h)		PCI-X Capability ID (07h)		Yes	No
F4h	PCI-X Bridge Status								Yes	No
F8h	PCI-X Upstream Split Transaction								Yes	No
FCh	PCI-X Downstream Split Transaction								Yes	No

**Notes:** \* Writable only when the Read-Only Registers Write Enable bit is set (HSSRRC[7]=1; PCI:9Ch). Refer to the individual register descriptions to determine which bits are writable.

Refer to the individual register descriptions to determine which bits are writable.

## 6.1.1 PCI Type 1 Header Registers—Transparent Mode

Register 6-1. (PCIIDR; PCI:00h) PCI Configuration ID

Bit	Description	Read	Write	Value after Reset
15:0	<b>Vendor ID.</b> Identifies PCI 6540 manufacturer. Defaults to the PCI-SIG-issued PLX Vendor ID (3388h), if a blank or no serial EEPROM is present.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	3388h
31:16	<b>Device ID.</b> Identifies the particular device. Defaults to PLX PCI 6540 part number (0028h), if a blank or no serial EEPROM is present. <b>Note:</b> In Non-Transparent mode, defaults to 0029h.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0028h

Register 6-2. (PCICR; PCI:04h) Primary PCI Command

Bit	Description	Read	Write	Value after Reset
0	<b>I/O Space Enable.</b> Controls bridge response to I/O accesses on primary interface. Values: 0 = Ignores I/O transactions 1 = Enables response to I/O transactions	Yes	Yes	0
1	<b>Memory Space Enable.</b> Controls bridge response to Memory access on primary interface. Values: 0 = Ignores Memory transactions 1 = Enables response to Memory transactions	Yes	Yes	0
2	<b>Master Enable.</b> Controls bridge ability to operate as a master on primary interface. Values: 0 = Does not initiate transactions on primary interface and disables response to Memory or I/O transactions on secondary interface 1 = Enables the bridge to operate as a master on primary interface	Yes	Yes	0
3	<b>Special Cycle Enable. Not Supported.</b>	Yes	No	0
4	<b>Memory Write and Invalidate Enable. Not Supported.</b>	Yes	No	0
5	<b>VGA Palette Snoop Enable.</b> Controls bridge response to VGA-compatible Palette accesses. Values: 0 = Ignores VGA Palette accesses on primary interface 1 = Enables response to VGA Palette writes on primary interface (I/O address AD[9:0]=3C6h, 3C8h, and 3C9h)	Yes	Yes	0
6	<b>Parity Error Response Enable.</b> Controls bridge response to Parity errors. Values: 0 = Ignores Parity errors 1 = Performs normal parity checking	Yes	Yes	0
7	<b>Wait Cycle Control.</b> If set to 1, the PCI 6540 performs address/data stepping.	Yes	Yes	1

Register 6-2. (PCICR; PCI:04h) Primary PCI Command (Continued)

Bit	Description	Read	Write	Value after Reset
8	<b>P_SERR# Enable.</b> Controls the primary System Error (P_SERR#) pin enable. Values: 0 = Disables P_SERR# driver 1 = Enables P_SERR# driver	Yes	Yes	0
9	<b>Fast Back-to-Back Enable.</b> Controls bridge ability to generate Fast Back-to-Back transactions to various devices on primary interface. Values: 0 = No Fast Back-to-Back transactions 1 = <b>Reserved</b> ; PCI 6540 does not generate Fast Back-to-Back cycles	Yes	Yes	0
15:10	<b>Reserved.</b>	Yes	No	0h

Register 6-3. (PCISR; PCI:06h) Primary PCI Status

Bit	Description	Read	Write	Value after Reset
3:0	<b>Reserved.</b>	Yes	No	0h
4	<b>New Capability Functions Support.</b> Writing 1 supports New Capabilities Functions. The New Capability Function ID is located at the PCI Configuration space offset, determined by the New Capabilities linked list pointer value at CAP_PTR; PCI:34h.	Yes	No	1
5	<b>66 MHz-Capable.</b> If set to 1, this device supports a 66 MHz PCI clock environment.	Yes	No	1
6	<b>UDF.</b> No User-Definable Features.	Yes	No	0
7	<b>Fast Back-to-Back Capable.</b> Fast Back-to-Back write capable on primary port.	Yes	No	0
8	<b>Data Parity Error Detected.</b> Set when the following conditions are met: <ul style="list-style-type: none"> <li>P_PERR# is asserted, and</li> <li>Command register Parity Error Response Enable bit is set (PCICR[6]=1; PCI:04h)</li> </ul> Writing 1 clears bit to 0.	Yes	Yes/Clr	0
10:9	<b>DEVSEL# Timing.</b> Reads as 01b to indicate PCI 6540 responds no slower than with medium timing.	Yes	No	01b
11	<b>Signaled Target Abort.</b> Set by a target device when a Target Abort cycle occurs. Writing 1 clears bit to 0.	Yes	Yes/Clr	0
12	<b>Received Target Abort.</b> Set to 1 by the PCI 6540 when transactions are terminated with Target Abort. Writing 1 clears bit to 0.	Yes	Yes/Clr	0
13	<b>Received Master Abort.</b> Set to 1 by the PCI 6540 when transactions are terminated with Master Abort. Writing 1 clears bit to 0.	Yes	Yes/Clr	0
14	<b>Signaled System Error.</b> Set when P_SERR# is asserted. Writing 1 clears bit to 0.	Yes	Yes/Clr	0
15	<b>Detected Parity Error.</b> Set when a Parity error is detected, regardless of the Parity Error Response Enable bit state (PCICR[6]=X; PCI:04h). Writing 1 clears bit to 0.	Yes	Yes/Clr	0

**Register 6-4. (PCIREV; PCI:08h) PCI Revision ID**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Revision ID.</b> PCI 6540 revision.	Yes	No	1h

**Register 6-5. (PCICCR; PCI:09h – 0Bh) PCI Class Code**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Register Level Programming Interface.</b> None defined.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0h
15:8	<b>Subclass Code.</b> PCI-to-PCI bridge (Transparent mode) or other bridge device (Non-Transparent mode).	Yes	Only if HSSRRC[7]=1; Serial EEPROM	Transparent mode=04h, Non-Transparent mode=80h
23:16	<b>Base Class Code.</b> Bridge device.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	06h

**Register 6-6. (PCICLSR; PCI:0Ch) PCI Cache Line Size**

Bit	Description	Read	Write	Value after Reset
7:0	<b>System Cache Line Size.</b> Specified in units of 32-bit words (Dwords). Only cache line sizes of a power of two are valid. Maximum value is 20h. For values greater than 20h, PCI 6540 operates as if PCICLSR is programmed with value of 08h. Used when terminating Memory Write and Invalidate transactions. Memory Read prefetching is controlled by the Prefetch Count registers.	Yes	Yes	0h

**Register 6-7. (PCILTR; PCI:0Dh) Primary PCI Bus Latency Timer**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Primary PCI Bus Latency Timer.</b> Specifies amount of time (in units of PCI Bus clocks) the PCI 6540, as a bus master, can burst data on the primary PCI Bus.	Yes	Yes	0h

Register 6-8. (PCIHTR; PCI:0Eh) PCI Header Type

Bit	Description	Read	Write	Value after Reset
6:0	<b>Configuration Layout Type.</b> Specifies register layout at offsets 10h to 3Fh in Configuration space. Header Type 0 is defined for PCI devices other than PCI-to-PCI bridges (Header Type 1) and Cardbus bridges (Header Type 2). <i>Note:</i> Default value is 0h in Non-Transparent mode.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	1h
7	<b>Multi-Function Device.</b> Value of 1 indicates multiple (up to eight) functions (logical devices), each containing its own, individually addressable Configuration space, 64 Dwords in size.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0

Register 6-9. (PCIBISTR; PCI:0Fh) PCI Built-In Self-Test

Bit	Description	Read	Write	Value after Reset
7:0	<b>Built-In Self-Test (BIST).</b> <i>Not Supported.</i>	Yes	No	0h

**Register 6-10. (PCIPBNO; PCI:18h) PCI Primary Bus Number**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Primary Bus Number.</b> Programmed with the PCI and/or PCI-X Bus number to which the primary bridge interface is connected.	Yes	Yes	0h

**Register 6-11. (PCISBNO; PCI:19h) PCI Secondary Bus Number**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Secondary Bus Number.</b> Programmed with the PCI and/or PCI-X Bus number to which the secondary bridge interface is connected.	Yes	Yes	0h

**Register 6-12. (PCISUBNO; PCI:1Ah) PCI Subordinate Bus Number**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Subordinate Bus Number.</b> Programmed with the PCI and/or PCI-X Bus Number with the highest number subordinate to the bridge.	Yes	Yes	0h

**Register 6-13. (PCISLTR; PCI:1Bh) Secondary PCI Bus Latency Timer**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Secondary PCI Bus Latency Timer.</b> Specifies the amount of time (in units of PCI Bus clocks) the PCI 6540, as a bus master, can burst data on the secondary PCI Bus.	Yes	Yes	0h

**Register 6-14. (PCIIOBAR; PCI:1Ch) I/O Base**

Bit	Description	Read	Write	Value after Reset
7:0	<p><b>I/O Base.</b> Specifies the I/O Address Range bits [15:12] base (Base Address bits [11:0] are assumed to be 0h).</p> <p>Used in conjunction with the I/O Limit, I/O Base Upper 16 Bits, and I/O Limit Upper 16 Bits registers (PCIOLMT; PCI:1Dh, PCIIOBARU16; PCI:30h, and PCIOLMTU16; PCI:32h, respectively) to specify a range of 32-bit addresses supported for PCI Bus I/O transactions.</p> <p>The lower four bits [3:0] are Read-Only and hardcoded to 0001b to indicate 32-bit I/O addressing support.</p>	Yes	Yes [7:4]	1h

**Register 6-15. (PCIOLMT; PCI:1Dh) I/O Limit**

Bit	Description	Read	Write	Value after Reset
7:0	<p><b>I/O Limit.</b> Specifies the upper address of the I/O Address Range bits [15:12] (Limit Address bits [11:0] are assumed to be FFFh).</p> <p>Used in conjunction with the I/O Base, I/O Base Upper 16 Bits, and I/O Limit Upper 16 Bits registers (PCIIOBAR; PCI:1Ch, PCIIOBARU16; PCI:30h, and PCIOLMTU16; PCI:32h, respectively) to specify a range of 32-bit addresses supported for PCI Bus I/O transactions.</p> <p>The lower four bits [3:0] are Read-Only and hardcoded to 0001b to indicate 32-bit I/O addressing support.</p>	Yes	Yes [7:4]	1h

Register 6-16. (PCISSR; PCI:1Eh) Secondary PCI Status

Bit	Description	Read	Write	Value after Reset
4:0	<b>Reserved.</b>	Yes	No	0h
5	<b>66 MHz-Capable.</b> If set to 1, the PCI 6540 supports a 66 MHz PCI clock environment.	Yes	No	1
6	<b>UDF.</b> No User-definable features.	Yes	No	0
7	<b>Fast Back-to-Back Capable.</b> Fast Back-to-Back write capable on secondary port. <b>Not Supported.</b>	Yes	No	0
8	<b>Data Parity Error Detected.</b> Set when the following conditions are met: <ul style="list-style-type: none"> <li>S_PERR# is asserted, and</li> <li>Command register Parity Error Response Enable bit is set (PCICR[6]=1; PCI:04h)</li> </ul> Writing 1 clears bit to 0.	Yes	Yes/Clr	0
10:9	<b>DEVSEL# Timing.</b> Reads as 01b to indicate PCI 6540 responds no slower than with medium timing	Yes	No	01b
11	<b>Signaled Target Abort.</b> Set by a target device when a Target Abort cycle occurs. Writing 1 clears bit to 0.	Yes	Yes/Clr	0
12	<b>Received Target Abort.</b> Set to 1 by the PCI 6540 when transactions are terminated with Target Abort. Writing 1 clears bit to 0.	Yes	Yes/Clr	0
13	<b>Received Master Abort.</b> Set to 1 by the PCI 6540 when transactions are terminated with Master Abort. Writing 1 clears bit to 0.	Yes	Yes/Clr	0
14	<b>Signaled System Error.</b> Set when S_SERR# is asserted. Writing 1 clears bit to 0.	Yes	Yes/Clr	0
15	<b>Detected Parity Error.</b> Set when a Parity error is detected, regardless of the Parity Error Response Enable bit state (PCICR[6]=X; PCI:04h). Writing 1 clears bit to 0.	Yes	Yes/Clr	0



Register 6-17. (PCIMBAR; PCI:20h) Memory Base

Bit	Description	Read	Write	Value after Reset
15:0	<p><b>Memory Base.</b> Specifies the Memory-Mapped I/O Address Range bits [31:20] base. The upper 12 bits corresponding to [31:20] are writable. The lower 20 Address bits [19:0] are assumed to be 0h.</p> <p>Used in conjunction with the Memory Limit register (PCIMLMT; PCI:22h) to specify a range of 32-bit addresses supported for PCI Bus Memory-Mapped I/O transactions.</p> <p>The lower four bits [3:0] are Read-Only and hardcoded to 0h.</p>	Yes	Yes [15:4]	0h

Register 6-18. (PCIMLMT; PCI:22h) Memory Limit

Bit	Description	Read	Write	Value after Reset
15:0	<p><b>Memory Limit.</b> Specifies the Memory-Mapped I/O Address Range bits [31:20] upper address. The upper 12 bits corresponding to [31:20] are writable. The lower 20 Address bits [19:0] are assumed to be F_FFFFh.</p> <p>Used in conjunction with the Memory Base register (PCIMBAR; PCI:20h) to specify a range of 32-bit addresses supported for PCI Bus Memory-Mapped I/O transactions.</p> <p>The lower four bits [3:0] are Read-Only and hardcoded to 0h.</p>	Yes	Yes [15:4]	0h

**Register 6-19. (PCIPMBAR; PCI:24h) Prefetchable Memory Base**

Bit	Description	Read	Write	Value after Reset
15:0	<p><b>Prefetchable Memory Base.</b> Specifies the Prefetchable Memory-Mapped Address Range bits [31:20] base. The upper 12 bits corresponding to [31:20] are writable. The lower 20 Address bits [19:0] are assumed to be 0h. Used in conjunction with the Prefetchable Memory Limit, Prefetchable Memory Base Upper 32 Bits, and Prefetchable Memory Limit Upper 32 Bits registers (PCIPMLMT; PCI:26h, PCIPMBARU32; PCI:28h, and PCIPMLMTU32; PCI:2Ch, respectively) to specify a range of 64-bit addresses supported for Prefetchable Memory transactions on the PCI Bus. The lower four Read-Only bits are hardcoded to 1h, indicating 64-bit address support.</p>	Yes	Yes [15:4]	1h

**Register 6-20. (PCIPMLMT; PCI:26h) Prefetchable Memory Limit**

Bit	Description	Read	Write	Value after Reset
15:0	<p><b>Prefetchable Memory Limit.</b> Specifies the Prefetchable Memory-Mapped Address Range bits [31:20] upper address. The upper 12 bits corresponding to [31:20] are writable. The lower 20 Address bits [19:0] are assumed to be F_FFFFh. Used in conjunction with the Prefetchable Memory Base, Prefetchable Memory Base Upper 32 Bits, and Prefetchable Memory Limit Upper 32 Bits registers (PCIPMBAR; PCI:24h, PCIPMBARU32; PCI:28h, and PCIPMLMTU32; PCI:2Ch, respectively) to specify a range of 64-bit addresses supported for Prefetchable Memory transactions on the PCI Bus. The lower four Read-Only bits are hardcoded to 1h, indicating 64-bit address support.</p>	Yes	Yes [15:4]	1h

**Register 6-21. (PCIPMBARU32; PCI:28h) Prefetchable Memory Base Upper 32 Bits**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Prefetchable Memory Base Upper 32 Bits.</b> Specifies the Prefetchable Memory-Mapped Address Range bits [63:32] base. The lower 20 Address bits [19:0] are assumed to be 0h. Used in conjunction with the Prefetchable Memory Base, Prefetchable Memory Limit, and Prefetchable Memory Limit Upper 32 Bits registers (PCIPMBAR; PCI:24h, PCIPMLMT; PCI:26h, and PCIPMLMTU32; PCI:2Ch, respectively) to specify a range of 64-bit addresses supported for Prefetchable Memory transactions on the PCI Bus.	Yes	Yes	0h

**Register 6-22. (PCIPMLMTU32; PCI:2Ch) Prefetchable Memory Limit Upper 32 Bits**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Prefetchable Memory Limit Upper 32 Bits.</b> Specifies the Prefetchable Memory-Mapped Address Range bits [31:20] upper address. The upper 12 bits corresponding to bits [31:20] are writable. The lower 20 Address bits [19:0] are assumed to be F_FFFFh. Used in conjunction with the Prefetchable Memory Base, Prefetchable Memory Limit, and Prefetchable Memory Base Upper 32 Bits registers (PCIPMBAR; PCI:24h, PCIPMLMT; PCI:26h, and PCIPMBARU32; PCI:28h, respectively) to specify a range of 64-bit addresses supported for Prefetchable Memory transactions on the PCI Bus.	Yes	Yes	0h

**Register 6-23. (PCIIOBARU16; PCI:30h) I/O Base Upper 16 Bits**

Bit	Description	Read	Write	Value after Reset
15:0	<b>I/O Base Upper 16 Bits.</b> Specifies the I/O Address Range bits [31:16] base. Base Address bits [11:0] are assumed to be 0h. Used in conjunction with the I/O Base, I/O Limit, and I/O Limit Upper 16 Bits registers (PCIIOBAR; PCI:1Ch, PCIIOGMT; PCI:1Dh, and PCIIOGMTU16; PCI:32h, respectively) to specify a range of 32-bit addresses supported for PCI Bus I/O transactions.	Yes	Yes	0h

**Register 6-24. (PCIIOGMTU16; PCI:32h) I/O Limit Upper 16 Bits**

Bit	Description	Read	Write	Value after Reset
15:0	<b>I/O Limit Upper 16 Bits.</b> Specifies the I/O Address Range bits [31:16] upper address. Limit Address bits [11:0] are assumed to be FFFh. Used in conjunction with the I/O Base, I/O Limit, and I/O Base Upper 16 Bits registers (PCIIOBAR; PCI:1Ch, PCIIOGMT; PCI:1Dh, and PCIIOBARU16; PCI:30h, respectively) to specify a range of 32-bit addresses supported for PCI Bus I/O transactions.	Yes	Yes [7:4]	0h

Register 6-25. (CAP\_PTR; PCI:34h) New Capability Pointer

Bit	Description	Read	Write	Value after Reset
7:0	<b>New Capability Pointer.</b> Provides an offset into PCI Configuration space for the Power Management capability location in the New Capabilities Linked List.	Yes	No	DCh
31:8	<b>Reserved.</b>	Yes	No	0h

Register 6-26. (PCIIPR; PCI:3Dh) PCI Interrupt Pin

Bit	Description	Read	Write	Value after Reset
7:0	<b>Interrupt Pin.</b> Indicates which interrupt pin the PCI 6540 uses. The following value is valid: 0h = No Interrupt pin	Yes	No	0h

Register 6-27. (BCNTRL; PCI:3Eh) Bridge Control

Bit	Description	Read	Write	Value after Reset
0	<b>Parity Error Response Enable.</b> Controls bridge response to Parity errors on secondary interface. Values: 0 = Ignores Address and Data Parity errors on secondary interface 1 = Enables Parity error reporting and detection on secondary interface	Yes	Yes	0
1	<b>S_SERR# Enable.</b> Controls forwarding of S_SERR# to primary interface. Values: 0 = Disables S_SERR# forwarding to primary 1 = Enables S_SERR# forwarding to primary	Yes	Yes	0
2	<b>ISA Enable.</b> Controls bridge response to ISA I/O addresses, which is limited to the first 64 KB. Values: 0 = Forwards I/O addresses in the range defined by the I/O Base and I/O Limit registers (PCIIOBAR; PCI:1Ch and PCIIO LMT; PCI:1Dh, respectively). 1 = Blocks forwarding of ISA I/O addresses in the range defined by the I/O Base and I/O Limit registers in the first 64 KB of I/O space that address the last 768 bytes in each 1-KB block. Secondary I/O transactions are forwarded upstream, if the address falls within the last 768 bytes in each 1-KB block. Command Configuration register Master Enable bit must also be set (PCICR[2]=1; PCI:04h) to enable ISA.	Yes	Yes	0
3	<b>VGA Enable.</b> Controls bridge response to VGA-compatible addresses. Values: 0 = Does not forward VGA-compatible Memory nor I/O addresses from primary to secondary 1 = Forwards VGA-compatible Memory and I/O address from primary to secondary, regardless of other settings	Yes	Yes	0

Register 6-27. (BCNTRL; PCI:3Eh) Bridge Control (Continued)

Bit	Description	Read	Write	Value after Reset
4	<b>Reserved.</b>	Yes	No	0
5	<b>Master Abort Mode.</b> Controls bridge behavior in response to Master Aborts on secondary interface. Values: 0 = Does not report Master Aborts (return FFFF_FFFFh on reads nor discard data on writes). 1 = Reports Master Aborts by signaling Target Abort. If the Master Abort is the result of a primary-to-secondary Posted Write cycle, P_SERR# is asserted (PCICR[8]=1; PCI:04h).  <b>Note:</b> During Lock cycles, PCI 6540 ignores this bit, and completes the cycle as a Target Abort.	Yes	Yes	0
6	<b>Secondary Reset.</b> Forces S_RSTOUT# assertion on secondary interface. Values: 0 = Does not force S_RSTOUT# assertion 1 = Forces S_RSTOUT# assertion	Yes	Yes	0
7	<b>Fast Back-to-Back Enable.</b> Controls bridge ability to generate Fast Back-to-Back transactions to various devices on secondary interface. Values: 0 = No Fast Back-to-Back transaction 1 = <b>Reserved</b> ; PCI 6540 does not generate Fast Back-to-Back cycles	Yes	Yes	0
8	<b>Primary Master Timeout.</b> Sets the maximum number of PCI clocks for an initiator on the primary bus to repeat the Delayed transaction request. Values: 0 = Timeout after $2^{15}$ PCI clocks 1 = Timeout after $2^{10}$ PCI clocks	Yes	Yes	0
9	<b>Secondary Master Timeout.</b> Sets the maximum number of PCI clocks for an initiator on the secondary bus to repeat the Delayed transaction request. Values: 0 = Timeout after $2^{15}$ PCI clocks 1 = Timeout after $2^{10}$ PCI clocks	Yes	Yes	0
10	<b>Master Timeout Status.</b> Set to 1 when primary or secondary Master Timeout occurs. Writing 1 clears bit to 0.	Yes	Yes/Clr	0
11	<b>Master Timeout P_SERR# Enable.</b> Enable P_SERR# assertion during Master Timeout. Values: 0 = P_SERR# not asserted on Master Timeout 1 = P_SERR# asserted on primary or secondary Master Timeout	Yes	Yes	0
15:12	<b>Reserved.</b>	Yes	No	0h

## 6.1.2 Device-Specific Registers—Transparent Mode

### 6.1.2.1 Chip, Diagnostic, and Arbiter Control Registers

Register 6-28. (CCNTL; PCI:40h) Chip Control

Bit	Description	Read	Write	Value after Reset
0	<b>Reserved.</b>	Yes	No	0
1	<b>Memory Write Disconnect Control.</b> Controls when PCI 6540, as a target, Disconnects Memory transactions. Values: 0 = Disconnects on queue full or on a 4-KB boundary 1 = Disconnects on a cache line boundary, when the queue fills, or on a 4-KB boundary	Yes	Yes	0
2	<b>Private Memory Enable (Transparent Mode).</b> The Memory space can be programmed using the Private Memory Base and Limit registers (PVTMBAR; PCI:6Ch and PVTLMT; PCI:6Eh, respectively). If the Limit is smaller than the Base, the Private Memory space is disabled regardless of bit setting. When enabled, the primary port cannot access primary and secondary Private Memory space through the bridge and secondary ports <b>do not</b> respond to Memory cycles addressing the Private Memory space. Resets to the value presented on the PRV_DEV input pin. After reset, bit can be reprogrammed. Values: 0 = Disables Private Memory block 1 = Enables Private Memory block	Yes	Yes	PRV_DEV
3	<b>Private Device Enable.</b> PCI 6540 can reroute secondary IDSELs using S_AD[23:16] for private devices. A Type 1 Configuration access on the primary bus (which would normally result in the assertion of an IDSEL connected to S_AD[23:16]) is routed to S_AD24. If there is no device on S_AD24, the rerouted Type 1 Configuration cycles result in a Master Abort. Re-routing allows S_AD[23:16] to be used for secondary private devices. This mechanism has no effect in Non-Transparent mode. Resets to the value presented in the PRV_DEV input pin. After reset, bit can be reprogrammed. Values: 0 = Does not re-route IDSEL assertions 1 = Enables the rerouting of the secondary IDSELs S_AD[23:16] to S_AD24	Yes	Yes	PRV_DEV
4	<b>Secondary Bus Prefetch Disable.</b> Controls PCI 6540 ability to prefetch during downstream Memory Read transactions. Values: 0 = Prefetches and does not forward Byte Enables during Memory Read transactions. 1 = Requests only 1 Dword from the target during Memory Read transactions and forwards Byte Enables. PCI 6540 returns a Target Disconnect to the requesting master on the first Data transfer. Memory Read Line and Memory Read Multiple transactions remain prefetchable.	Yes	Yes	0
7:5	<b>Reserved.</b>	Yes	No	000b

Register 6-29. (DCNTRL; PCI:41h) Diagnostic Control

Bit	Description	Read	Write	Value after Reset
0	<b>Chip Reset.</b> Chip and secondary bus reset. Setting bit activates full chip reset, asserts S_RSTOUT#, and forces the Bridge Control register Secondary Reset bit to be set (BCNTRL[6]=1; PCI:3Eh). After resetting the PCI 6540 registers, bit is cleared; however, BCNTRL[6] remains set to 1. Writing 0 has no effect.	Yes	Yes	0
2:1	<b>Test Mode.</b>	Yes	Yes	00b
3	<b>Secondary Reset Output Mask. <i>Not Supported.</i></b>	Yes	No	0
7:4	<b>Reserved.</b>	Yes	No	0h

Register 6-30. (ACNTRL; PCI:42h) Arbiter Control

Bit	Description	Read	Write	Value after Reset
7:0	<b>Arbiter Control.</b> Each bit controls whether a secondary bus master is assigned to the high- or low-priority group. Bits correspond to request inputs S_REQ[7:0]#, respectively. Value of 1h assigns the bus master to the high-priority group.	Yes	Yes	0h
8	<b>Reserved.</b>	Yes	Yes	0
9	<b>PCI 6540 Priority.</b> Defines whether PCI 6540 secondary port is in the high- or low-priority group. 0 = Low-priority group 1 = High-priority group	Yes	Yes	1
11:10	<b>Reserved.</b>	Yes	No	00b
12	<b>Primary Port Ordering Rule. <i>Reserved and must be set to 0.</i></b>	Yes	Yes	0
13	<b>Secondary Port Ordering Rule. <i>Reserved and must be set to 0.</i></b>	Yes	Yes	0
14	<b>Upstream 64-Bit Cycle Control. <i>Reserved and must be set to 0.</i></b>	Yes	Yes	0
15	<b>Downstream 64-Bit Cycle Control. <i>Reserved and must be set to 0.</i></b>	Yes	Yes	0

### 6.1.2.2 Primary Flow-Through Control Register

Register 6-31. (PFTCR; PCI:44h) Primary Flow-Through Control

Bit	Description	Read	Write	Value after Reset
2:0	<p><b>Primary Posted Write-Completion Wait Count.</b> Maximum number of clocks the PCI 6540 waits for Posted Write data from the initiator, if delivering Write data in Flow-Through mode, with the Internal Post Write queues almost empty. If the count is exceeded, without additional data from the initiator, the cycle to the target is terminated and later completed. Values:</p> <p>000b = De-asserts S_IRDY# and waits seven clocks for data on the primary bus before terminating cycle (same as 111b)</p> <p>001b = De-asserts S_IRDY# and waits one clock for data on the primary bus before terminating cycle</p> <p>...</p> <p>111b = De-asserts S_IRDY# and waits seven clocks for data on the primary bus before terminating cycle</p>	Yes	Yes; Serial EEPROM	000b
3	<i>Reserved.</i>	Yes	No	0
6:4	<p><b>Primary Delayed Read Completion Wait Count.</b> Maximum number of clocks the PCI 6540 waits for Delayed Read data from the target if returning Read data in Flow-Through mode, and the Internal Delayed Read queue is almost full. If the count is exceeded without additional space in the queue, the cycle to the target is terminated, and completed when the initiator Retries the remainder of the cycle. Values:</p> <p>000b = De-asserts S_IRDY# and waits seven clocks for further data to be transferred to the primary bus before terminating cycle (same as 111b)</p> <p>001b = De-asserts S_IRDY# and waits one clock for additional space to be transferred to the primary bus before terminating cycle</p> <p>...</p> <p>111b = De-asserts S_IRDY# and waits seven clocks for additional space to be transferred to the primary bus before terminating cycle</p>	Yes	Yes; Serial EEPROM	000b
7	<i>Reserved.</i>	Yes	No	0



## 6.1.2.3 Timeout Control Register

Register 6-32. (TOCNTRL; PCI:45h) Timeout Control

Bit	Description	Read	Write	Value after Reset
2:0	<b>Maximum Retry Counter Control.</b> Controls the maximum number of times the PCI 6540 Retries a cycle before signaling a timeout. Timeout applies to Read/Write Retries and can be enabled to trigger SERR# on the primary or secondary port, depending on the SERR# events enabled. Maximum number of Retries to timeout: 000b = $2^{24}$ 001b = $2^{18}$ 010b = $2^{12}$ 011b = $2^6$ 111b = $2^0$	Yes	Yes; Serial EEPROM	000b
3	<b>Reserved.</b>	Yes	No	0
5:4	<b>Primary Master Timeout Divider.</b> Provides additional options for the primary Master Timeout. In addition to its original setting in the Bridge Control register (BCNTRL[8]; PCI:3Eh), the Timeout Counter can optionally be divided by up to 256: 00b = Counter—Primary Master Timeout / 1 01b = Timeout Counter—Primary Master Timeout / 8 10b = Timeout Counter—Primary Master Timeout / 16 11b = Timeout Counter—Primary Master Timeout / 256 BCNTRL[8] can set the primary Master Timeout to 32K (default) or 1K Clock cycles.	Yes	Yes; Serial EEPROM	00b
7:6	<b>Secondary Master Timeout Divider.</b> Provides additional options for the secondary Master Timeout. In addition to its original setting in the Bridge Control register (BCNTRL[9]; PCI:3Eh), the Timeout Counter can optionally be divided by up to 256: 00b = Counter—Secondary Master Timeout / 1 01b = Timeout Counter—Secondary Master Timeout / 8 10b = Timeout Counter—Secondary Master Timeout / 16 11b = Timeout Counter—Secondary Master Timeout / 256 BCNTRL[9] can set the secondary Master Timeout to 32K (default) or 1K Clock cycles.	Yes	Yes; Serial EEPROM	00b

### 6.1.2.4 Miscellaneous Options Register

Register 6-33. (MSCOPT; PCI:46h) Miscellaneous Options

Bit	Description	Read	Write	Value after Reset
0	<b>Write Completion Wait for PERR#.</b> If set to 1, PCI 6540 waits for target PERR# status before completing a Delayed Write transaction to the initiator.	Yes	Yes; Serial EEPROM	0
1	<b>Read Completion Wait for PAR.</b> If set to 1, PCI 6540 waits for target PAR status before completing a Delayed Read transaction to the initiator.	Yes	Yes; Serial EEPROM	0
2	<b>DRT Out-of-Order Enable.</b> If set to 1, PCI 6540 may return Delayed Read transactions in a different order than requested. Otherwise, Delayed Read transactions are returned in the same order as requested.	Yes	Yes; Serial EEPROM	0
3	<b>Generate Parity Enable.</b> Values: 0 = Passes along the cycle PAR/PAR64, as stored in the internal buffers 1 = PCI 6540, as a master, generates PAR/PAR64 to cycles traveling across the bridge	Yes	Yes; Serial EEPROM	0

Register 6-33. (MSCOPT; PCI:46h) Miscellaneous Options (Continued)

Bit	Description	Read	Write	Value after Reset
6:4	<b>Address Step Control.</b> During Type 0 Configuration cycles, PCI 6540 drives the address for the number of clocks specified by these bits, before asserting FRAME#. Defaults to 001b in Conventional PCI mode. In PCI-X mode, these bits have no effect and address stepping is hardcoded to four clocks. Values: 000b = Concurrently asserts FRAME# and drives the address on the bus 001b = Asserts FRAME# one clock after driving the address on the bus ... 111b = Asserts FRAME# seven clocks after driving the address on the bus	Yes	Yes; Serial EEPROM	001b
8:7	<b>Reserved.</b>	Yes	No	00b
9	<b>Prefetch Early Termination.</b> Values: 0 = Terminates prefetching at the Initial Prefetch Count if Flow Through is not achieved, and another Prefetching Read cycle is accepted by the PCI 6540 1 = Completes prefetching as programmed by the Prefetch Count registers, regardless of other outstanding prefetchable reads in the Transaction queue	Yes	Yes; Serial EEPROM	0
10	<b>Read Minimum Enable.</b> If set to 1, PCI 6540 only initiates Read cycles if there is sufficient space available in the FIFO as required by the Prefetch Count registers.	Yes	Yes; Serial EEPROM	0
15, 11	<b>Force 64-Bit Control.</b> If set and the target supports 64-bit transfers, 32-bit Prefetchable reads or 32-bit Posted Memory Write cycles on one side are converted to 64-bit cycles on completion at the target bus. If set to 00b, cycles are not converted. Values: 00b = Disable (default) 01b = Convert to 64-bit command on both ports 10b = Convert to 64-bit command on secondary port 11b = Convert to 64-bit command on primary port	Yes	Yes; Serial EEPROM	0
12	<b>Memory Write and Invalidate Control.</b> Values: 0 = Retries Memory Write and Invalidate commands if there is insufficient space for one cache line of data in the internal queues. 1 = Passes Memory Write and Invalidate commands if there are one or more cache lines of FIFO space available. If there is insufficient space, completes as a Memory Write cycle.	Yes	Yes; Serial EEPROM	0
13	<b>Primary Lock Enable.</b> If set to 1, PCI 6540 follows the LOCK protocol on primary interface; otherwise, LOCK is ignored.	Yes	Yes; Serial EEPROM	0
14	<b>Secondary Lock Enable.</b> If set to 1, PCI 6540 follows the LOCK protocol on secondary interface; otherwise, LOCK is ignored.	Yes	Yes; Serial EEPROM	0

### 6.1.2.5 Prefetch Control Registers

Registers 48h to 4Eh are the Flow-Through Prefetch Control registers, which are used to fine-tune the PCI 6540 Memory Read prefetch behavior. (Refer to Section 18, “PCI Flow-Through Optimization,” for further details regarding these registers.) These registers apply only if there are one or more ports operating in Conventional PCI mode.

**Register 6-34. (PITLPCNT; PCI:48h) Primary Initial Prefetch Count**

Bit	Description	Read	Write	Value after Reset
0	<i>Reserved.</i>	Yes	No	0
2:1	<b>PCI-X Primary initial Prefetch Count.</b> When the primary bus is in PCI-X mode, these bits control the initial Prefetch Count on the primary bus during reads to Prefetchable Memory space. Value defines the cache line multiples for the initial Prefetch Count. Prefetch as follows: 00b = 1 cache line (size defined by PCICLSR; PCI:0Ch) 01b = 2 cache lines 10b = 4 cache lines 11b = 8 cache lines The Primary PCI-X 16 Cache Line Prefetch bit (PINPCNT[1]; PCI:4Ah) enables prefetch of 16 cache lines.	Yes	Yes; Serial EEPROM	00b
5:3	<b>PCI Primary initial Prefetch Count.</b> When the primary bus is in Conventional PCI mode, controls the Initial Prefetch Count on the primary bus during reads to Prefetchable Memory space. Prefetch as follows: 001b = 08h Dwords 010b = 10h Dwords 101b = 20h Dwords Other values are <i>Reserved.</i>	Yes	Yes; Serial EEPROM	001b
7:6	<i>Reserved.</i>	Yes	No	00b

Register 6-35. (SITLPCNT; PCI:49h) Secondary Initial Prefetch Count

Bit	Description	Read	Write	Value after Reset
0	<b>Reserved.</b>	Yes	No	0
2:1	<b>PCI-X Secondary Initial Prefetch Count.</b> When the secondary bus is in PCI-X mode, these bits control the initial Prefetch Count on the secondary bus during reads initiated from the primary port. Value defines the cache line multiples for the initial Prefetch Count. Prefetch as follows: 00b = 1 cache line (size defined by PCICLSR; PCI:0Ch) 01b = 2 cache lines 10b = 4 cache lines 11b = 8 cache lines The Secondary PCI-X 16 Cache Line Prefetch bit (SINPCNT[1]; PCI:4Bh) enables prefetch of 16 cache lines.	Yes	Yes; Serial EEPROM	00b
5:3	<b>PCI Secondary Initial Prefetch Count.</b> When the secondary bus is in Conventional PCI mode, these bits control the initial Prefetch Count on the secondary bus during reads initiated from the primary port. Prefetch as follows: 001b = 08h Dwords 010b = 10h Dwords 101b = 20h Dwords Other values are <b>Reserved.</b>	Yes	Yes; Serial EEPROM	001b
7:6	<b>Reserved.</b>	Yes	No	00b

Register 6-36. (PINPCNT; PCI:4Ah) Primary Incremental Prefetch Count

Bit	Description	Read	Write	Value after Reset
0	<b>Reserved.</b>	Yes	No	0
1	<b>Primary PCI-X 16 Cache Line Prefetch.</b> Value: 1 = Enables primary PCI-X port to prefetch 16 cache lines. When set, the PCI-X Primary Initial Prefetch Count bits are ignored (PITLPCNT[2:1]; PCI:48h).	Yes	Yes; Serial EEPROM	0
5:2	<b>Primary Incremental Prefetch Count.</b> Applies only to primary port Conventional PCI mode operation. These bits control the incremental read prefetch count. When an entry's remaining prefetch Dword count falls below this value, the bridge prefetches additional primary Incremental Prefetch Count Dwords. Value is specified as a multiple of 4 x Dwords. The register value must not exceed half the value programmed in the Primary Maximum Prefetch Count register (PMAXPNT; PCI:4Ch); otherwise, no incremental prefetch is performed. Prefetch as follows: 0000b = No incremental prefetch 0001b = 04h Dwords 0010b = 08h Dwords 0011b = 0Ch Dwords ... 1111b = 3Ch Dwords	Yes	Yes; Serial EEPROM	0000b
7:6	<b>Reserved.</b>	Yes	No	00b

Register 6-37. (SINCPNT; PCI:4Bh) Secondary Incremental Prefetch Count

Bit	Description	Read	Write	Value after Reset
0	<b>Reserved.</b>	Yes	No	0
1	<b>Secondary PCI-X 16 Cache Line Prefetch.</b> Value: 1 = Enables secondary PCI-X port to prefetch 16 cache lines. When set, the PCI-X Secondary Initial Prefetch Count bits are ignored (SITLPCNT[2:1]; PCI:49h).	Yes	Yes; Serial EEPROM	0
5:2	<b>Secondary Incremental Prefetch Count.</b> Applies only to secondary port Conventional PCI mode operation. These bits control the incremental read prefetch count. When an entry's remaining prefetch Dword count falls below this value, the bridge prefetches additional secondary incremental prefetch count Dwords. Value is specified as a multiple of 4 x Dwords. The register value must not exceed half the value programmed in the Secondary Maximum Prefetch Count register (SMAXPCNT; PCI:4Dh); otherwise, no incremental prefetch is performed. Prefetch as follows: 0000b = No incremental prefetch 0001b = 04h Dwords 0010b = 08h Dwords 0011b = 0Ch Dwords ... 1111b = 3Ch Dwords	Yes	Yes; Serial EEPROM	0000b
7:6	<b>Reserved.</b>	Yes	No	00b

**Register 6-38. (PMAXPcnt; PCI:4Ch) Primary Maximum Prefetch Count**

Bit	Description	Read	Write	Value after Reset
5:0	<p><b>Primary Maximum Prefetch Count.</b> Limits the cumulative maximum count of prefetchable Dwords allocated to one entry on the primary bus when Flow Through for that entry is not achieved. Value should be an even number. Bit 0 is Read-Only and always 0.</p> <p>Value is specified in Dwords, except if 0 value is programmed, which sets the Primary Maximum Prefetch Count to its maximum value of 256 bytes.</p> <p>This feature applies only to PCI-to-PCI bridging. A PCI Read cycle causes a PCI request for the Maximum Count data.</p>	Yes	Yes [5:1]; Serial EEPROM	20h
7:6	<i>Reserved.</i>	Yes	No	00b

**Register 6-39. (SMAXPcnt; PCI:4Dh) Secondary Maximum Prefetch Count**

Bit	Description	Read	Write	Value after Reset
5:0	<p><b>Secondary Maximum Prefetch Count.</b> Limits the cumulative maximum count of prefetchable Dwords allocated to one entry on the secondary bus when Flow Through for that entry is not achieved. Value should be an even number. Bit 0 is Read-Only and always 0.</p> <p>Value is specified in Dwords, except if 0 value is programmed, which sets the Secondary Maximum Prefetch Count to its maximum value of 256 bytes.</p> <p>This feature applies only to PCI-to-PCI bridging. A PCI Read cycle causes a PCI request for the Maximum Count data.</p>	Yes	Yes [5:1]; Serial EEPROM	20h
7:6	<i>Reserved.</i>	Yes	No	00b

### 6.1.2.6 Secondary Flow-Through Control Register

Register 6-40. (SFTCR; PCI:4Eh) Secondary Flow-Through Control

Bit	Description	Read	Write	Value after Reset
2:0	<b>Secondary Posted Write Completion Wait Count.</b> Maximum number of clocks the PCI 6540 waits for Posted Write data from the initiator if delivering Write data in Flow-Through mode and the Internal Post Write queues are almost empty. If the count is exceeded without additional data from the initiator, the cycle to the target is terminated and later completed. Values: 000b = De-asserts P_IRDY# and waits seven clocks for data on the secondary bus, before terminating cycle (same as 111b) 001b = De-asserts P_IRDY# and waits one clock for data on the secondary bus, before terminating cycle ... 111b = De-asserts P_IRDY# and waits seven clocks for data on the secondary bus, before terminating cycle	Yes	Yes; Serial EEPROM	000b
3	<b>Reserved.</b>	Yes	No	0
6:4	<b>Secondary Delayed Read Completion Wait Count.</b> Maximum number of clocks the PCI 6540 waits for Delayed Read data from the target, if returning Read data in Flow-Through mode and the Internal Delayed Read queue is almost full. If the count is exceeded without additional space in the queue, the cycle to target is terminated, and completed when initiator Retries the remainder of the cycle. Values: 000b = De-asserts P_IRDY# and waits seven clocks for additional space to be transferred to the secondary bus before terminating cycle (same as 111b) 001b = De-asserts P_IRDY# and waits one clock for additional space to be transferred to the secondary bus before terminating cycle ... 111b = De-asserts P_IRDY# and waits seven clocks for additional space to be transferred to the secondary bus before terminating cycle	Yes	Yes; Serial EEPROM	000b
7	<b>Reserved.</b>	Yes	No	0



## 6.1.2.7 Buffer and Internal Arbiter Control Registers

Register 6-41. (BUFCR; PCI:4Fh) Buffer Control

Bit	Description	Read	Write	Value after Reset
0	<b>Reserved.</b>	Yes	No	0
1	<b>Smart Prefetch Enable.</b> The amount of data prefetched is defined in the Maximum Prefetch Count registers (PMAXPNT; PCI:4Ch and SMAXPNT; PCI:4Dh). Values after a prefetch command: 0 = Remaining prefetched data is discarded upon completion of the current Read Command. 1 = Remaining prefetched data is <b>not</b> discarded, but remains available for the next Read Command with consecutive address. The prefetched data is only discarded upon a timeout. The timeout period can be programmed using the Smart Prefetch Timeout bits (BUFCR[6:5]; PCI:4Fh).	Yes	Yes	0
2	<b>Split FIFO Enable.</b> Buffer Split for individual data entries. Values: 0 = Entire FIFO shared among entries and the FIFO is undedicated 1 = FIFO divided into four equal parts, to be dedicated to each entry for Split Completions	Yes	Yes	0
4:3	<b>Reserved.</b>	Yes	No	00b
6:5	<b>Smart Prefetch Timeout.</b> Smart Prefetch Timeout affects only PCI-to-PCI, PCI-to-PCI-X, or PCI-X-to-PCI bridging applications. Prefetches cannot cross the 4-KB Address boundary. When Smart Prefetch is enabled, the prefetched data is only discarded upon a timeout. The timeout periods available are after: 00b = 32 PCI clocks 01b = 64 PCI clocks 10b = 128 PCI clocks 11b = 256 PCI clocks	Yes	Yes	11b
7	<b>Reserved.</b>	Yes	No	0

**Note:** Smart\_Prefetch settings apply only to Transparent mode PCI-to-PCI, PCI-to-PCI-X, or PCI-X-to-PCI bridging, where the initiator issues a PCI Burst Prefetchable read.

Register 6-42. (IACNTRL; PCI:50h) Internal Arbiter Control

Bit	Description	Read	Write	Value after Reset
0	<b>Low-Priority Group Fixed Arbitration.</b> If set to 1, the low-priority group uses fixed priority arbitration; otherwise, rotating priority arbitration is used.	Yes	Yes	0
1	<b>Low-Priority Group Arbitration Order.</b> Valid only when the low-priority arbitration group is set to a fixed arbitration scheme. Values: 0 = Priority decreases with bus master number. ( <i>For example, assuming Master 2 is set as the highest priority master, Master 3 retains higher priority than Master 4.</i> ) 1 = Priority increases with bus master number. ( <i>For example, assuming Master 2 is set as the highest priority master, Master 4 retains higher priority than Master 3.</i> ) This order is relative to the master with the highest priority for this group, as specified in IACNTRL[7:4].	Yes	Yes	0
2	<b>High-Priority Group Fixed Arbitration.</b> If set to 1, the high-priority group uses the fixed priority arbitration; otherwise, rotating priority arbitration is used.	Yes	Yes	0
3	<b>High-Priority Group Arbitration Order.</b> Valid only when the high-priority arbitration group is set to a fixed arbitration scheme. Values: 0 = Priority decreases with bus master number. ( <i>For example, assuming Master 2 is set as the highest priority master, Master 3 retains higher priority than Master 4.</i> ) 1 = Priority increases with bus master number. ( <i>For example, assuming Master 2 is set as the highest priority master, Master 4 retains higher priority than Master 3.</i> ) This order is relative to the master with the highest priority for this group, as specified in IACNTRL[11:8].	Yes	Yes	0
7:4	<b>Highest Priority Master in Low-Priority Group.</b> Controls which master in the low-priority group retain the highest priority. Valid only if the group uses the fixed arbitration scheme. Values: 0000b = Master 0 retains highest priority 0001b = Master 1 retains highest priority ... 1000b = PCI 6540 retains highest priority 1001b – 1111b = <b>Reserved</b>	Yes	Yes	0000b
11:8	<b>Highest Priority Master in High-Priority Group.</b> Controls which master in the high-priority group retains the highest priority. Valid only if the group uses the fixed arbitration scheme. Values: 0000b = Master 0 retains highest priority 0001b = Master 1 retains highest priority ... 1000b = PCI 6540 retains highest priority 1001b – 1111b = <b>Reserved</b>	Yes	Yes	0000b
15:12	<b>Bus Grant Parking Control.</b> Controls bus grant behavior during idle. Value: 0h = Indicates the last master granted is parked All other values are <b>Reserved</b> .	Yes	No	0h

## 6.1.2.8 Test and Serial EEPROM Registers

Register 6-43. (TEST; PCI:52h) Test

Bit	Description	Read	Write	Value after Reset
0	<b>Serial EEPROM Autoload Control.</b> If set to 1, disables serial EEPROM autoload.	Yes	Yes	0
1	<b>Fast Serial EEPROM Autoload.</b> If set to 1, speeds up serial EEPROM autoload.	Yes	Yes	0
2	<b>Serial EEPROM Autoload Status.</b> Serial EEPROM autoload status is set to 1 during autoload.	Yes	No	Status of Serial EEPROM Autoload
3	<b>S_PLL_TEST.</b> When P_CLKOE input pin is set to 1 and this bit is set to 1, S_CLKO4 (derived from S_CLKIN) is divided by 4.	Yes	Yes	0
4	<b>DEV64#.</b> Reflects DEV64# pin status.	Yes	No	DEV64#
5	<b>S_CFN#.</b> Reflects S_CFN# pin status.	Yes	No	S_CFN#
6	<b>TRANS#.</b> Reflects TRANS# pin status.	Yes	No	TRANS#
7	<b>U_MODE.</b> Reflects U_MODE pin status.	Yes	No	U_MODE

Register 6-44. (EEPCNTRL; PCI:54h) Serial EEPROM Control

Bit	Description	Read	Write	Value after Reset
0	<b>Start.</b> Starts serial EEPROM Read or Write cycle. Bit is cleared when serial EEPROM load completes.	Yes	Yes	0
1	<b>Serial EEPROM Command.</b> Controls commands sent to the serial EEPROM. Values: 0 = Read 1 = Write	Yes	Yes	0
2	<b>Serial EEPROM Error.</b> Set to 1 if serial EEPROM ACK was not received during serial EEPROM cycle.	Yes	No	—
3	<b>Serial EEPROM Autoload Successful.</b> Set to 1 if serial EEPROM autoload occurred successfully after reset, with appropriate Configuration registers loaded with the values programmed in the serial EEPROM. If 0, the serial EEPROM autoload was unsuccessful or disabled.	Yes	No	—
5:4	<b>Reserved.</b>	Yes	No	00b
7:6	<b>Serial EEPROM Clock Rate.</b> Controls the serial EEPROM clock frequency. The serial EEPROM clock is derived from the primary PCI clock. Values: 00b = PCLK / 1024 01b = PCLK / 512 10b = PCLK / 128 11b = PCLK / 16 (Test mode)	Yes	Yes	01b

Register 6-45. (EEPADDR; PCI:55h) Serial EEPROM Address

Bit	Description	Read	Write	Value after Reset
0	<i>Reserved.</i>	Yes	No	—
7:1	<b>Serial EEPROM Address.</b> Word address for the serial EEPROM cycle.	Yes	Yes	—

Register 6-46. (EEPDATA; PCI:56h) Serial EEPROM Data

Bit	Description	Read	Write	Value after Reset
15:0	<b>Serial EEPROM Data.</b> Contains data to be written to the serial EEPROM. During reads, contains data received from the serial EEPROM after a Read cycle completes.	Yes	Yes	—

## 6.1.2.9 Timer Registers

Register 6-47. (TMRCNTRL; PCI:61h) Timer Control

Bit	Description	Read	Write	Value after Reset
0	<b>Timer Enable.</b> Set to start measurement of the approximate bus frequency on the primary or secondary interface. By default, bit is 0, and must be set to 1 to start the measurement. When set to 0 and then to 1, PCI 6540 starts counting up until it reaches the set count period. During this counting period, PCI 6540 Timer Counter (TMRcnt; PCI:62h) counts the number of Timer Counter clocks.	Yes	Yes	0
2:1	<b>Timer Counter Clock Source Select.</b> Values: 00b = Primary PCI clock (P_CLKIN) 01b = Secondary PCI clock (S_CLKIN) 10b, 11b = <b>Reserved</b>	Yes	Yes	00b
3	<b>Timer Stop.</b> Timer stopped status bit. When the measurement is finished, this bit is set to 0, and then to 1. When starting a new measurement, this bit automatically restores to 0. Values: 0 = Timer running 1 = Timer stopped	Yes	No	0
5:4	<b>Count Period.</b> Values: 00b = 16 Reference clock high states 01b = 32 Reference clock high states 10b = 64 Reference clock high states 11b = 128 Reference clock high states	Yes	Yes	00b
7:6	<b>Reserved.</b>	Yes	No	00b

Register 6-48. (TMRcnt; PCI:62h) Timer Counter

Bit	Description	Read	Write	Value after Reset
15:0	<b>Timer Counter.</b> Automatically stops upon the count period setting in the Timer Control register (TMRCNTRL[7:4]; PCI:61h). This counter can be enabled by setting the Timer Enable bit (TMRCNTRL[0]; PCI:61h) first to 0, and then to 1.	Yes	No	0h

### 6.1.2.10 Primary System Error Event Register

Register 6-49. (PSERRED; PCI:64h) P\_SERR# Event Disable

Bit	Description	Read	Write	Value after Reset
0	<i>Reserved.</i>	Yes	No	0
1	<b>Posted Write Parity Error.</b> Controls PCI 6540 ability 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 bit is 0 and Command register P_SERR# Enable bit is set (PCICR[8]=1; PCI:04h).	Yes	Yes	0
2	<b>Posted Memory Write Non-Delivery.</b> Controls PCI 6540 ability to assert P_SERR# when it is unable to deliver Posted Write data after 2 <sup>24</sup> attempts [or programmed Maximum Retry count (TOCNTRL[2:0]; PCI:45h)]. P_SERR# is asserted if this event occurs when bit is 0 and Command register P_SERR# Enable bit is set (PCICR[8]=1; PCI:04h).	Yes	Yes	0
3	<b>Target Abort during Posted Write.</b> Controls PCI 6540 ability to assert P_SERR# when it receives a Target Abort while attempting to deliver Posted Write data. P_SERR# is asserted if this event occurs when bit is 0 and Command register P_SERR# Enable bit is set (PCICR[8]=1; PCI:04h).	Yes	Yes	0
4	<b>Master Abort on Posted Write.</b> Controls PCI 6540 ability to assert P_SERR# when it receives a Master Abort while attempting to deliver Posted Write data. P_SERR# is asserted if this event occurs when bit is 0 and Command register P_SERR# Enable bit is set (PCICR[8]=1; PCI:04h).	Yes	Yes	0
5	<b>Delayed Configuration or I/O Write Non-Delivery.</b> Controls PCI 6540 ability to assert P_SERR# when it is unable to deliver Delayed Write data after 2 <sup>24</sup> attempts [or programmed Maximum Retry count (TOCNTRL[2:0]; PCI:45h)]. P_SERR# is asserted if this event occurs when bit is 0 and Command register P_SERR# Enable bit is set (PCICR[8]=1; PCI:04h).	Yes	Yes	0
6	<b>Delayed Read-No Data from Target.</b> Controls PCI 6540 ability to assert P_SERR# when it is unable to transfer Read data from the target after 2 <sup>24</sup> attempts [or programmed Maximum Retry count (TOCNTRL[2:0]; PCI:45h)]. P_SERR# is asserted if this event occurs when bit is 0 and Command register P_SERR# Enable bit is set (PCICR[8]=1; PCI:04h).	Yes	Yes	0
7	<i>Reserved.</i>	Yes	No	0

## 6.1.2.11 GPIO[3:0] Registers

Register 6-50. (GPIOOD[3:0]; PCI:65h) GPIO[3:0] Output Data

Bit	Description	Read	Write	Value after Reset
3:0	<b>GPIO[3:0] Output Data Write 1 to Clear.</b> Writing 1 to these bits drives the corresponding signal low on the GPIO[3:0] bus, if the signal is programmed as an output. Writing 0 has no effect. Read returns the last written value.	Yes	Yes/Clr	0h
7:4	<b>GPIO[3:0] Output Data Write 1 to Set.</b> Writing 1 to these bits drives the corresponding signal high on the GPIO[3:0] bus, if the signal is programmed as an output. Writing 0 has no effect. Read returns the last written value.	Yes	Yes/Set High	0h

Register 6-51. (GPIOOE[3:0]; PCI:66h) GPIO[3:0] Output Enable

Bit	Description	Read	Write	Value after Reset
3:0	<b>GPIO[3:0] Output Enable Write 1 to Clear.</b> Writing 1 to these bits configures the corresponding signal on the GPIO[3:0] bus as an input. Writing 0 has no effect. Read returns the last written value.	Yes	Yes/Clr	0h
7:4	<b>GPIO[3:0] Output Enable Write 1 to Set.</b> Writing 1 to these bits configures the corresponding signal on the GPIO[3:0] bus as an output. Writing 0 has no effect. Read returns the last written value.	Yes	Yes/Set High	0h

Register 6-52. (GPIOID[3:0]; PCI:67h) GPIO[3:0] Input Data

Bit	Description	Read	Write	Value after Reset
3:0	<b>Reserved.</b>	Yes	No	0h
7:4	<b>GPIO[3:0] Input Data.</b> This Read-Only register reads GPIO[3:0] pins state. The state is updated on the PCI Clock cycle, following a change in GPIO[3:0] state.	Yes	No	—

### 6.1.2.12 Clock Control Register

Register 6-53. (CLKCNTRL; PCI:68h) Clock Control

Bit	Description	Read	Write	Value after Reset
1:0	<b>Clock 0 Disable.</b> If either bit is 0, S_CLKO0 is enabled. When both bits are 1, S_CLKO0 is disabled. Defaults to 00b if MSK_IN=1.	Yes	Yes	00b
3:2	<b>Clock 1 Disable.</b> If either bit is 0, S_CLKO1 is enabled. When both bits are 1, S_CLKO1 is disabled.	Yes	Yes	00b
5:4	<b>Clock 2 Disable.</b> If either bit is 0, S_CLKO2 is enabled. When both bits are 1, S_CLKO2 is disabled.	Yes	Yes	00b
7:6	<b>Clock 3 Disable.</b> If either bit is 0, S_CLKO3 is enabled. When both bits are 1, S_CLKO3 is disabled.	Yes	Yes	00b
8	<b>Clock 4 Disable.</b> If 0, S_CLKO4 is enabled. When 1, S_CLKO4 is disabled.	Yes	Yes	0
15:9	<b>Reserved.</b>	Yes	No	0h

### 6.1.2.13 Primary System Error Status Register

Register 6-54. (PSERRSR; PCI:6Ah) P\_SERR# Status

Bit	Description	Read	Write	Value after Reset
0	<b>Address Parity Error.</b> P_SERR# is asserted because an Address Parity error occurred on either side of the bridge.	Yes	Yes/Clr	0
1	<b>Posted Write Data Parity Error.</b> P_SERR# is asserted because a Posted Write Data Parity error occurred on the target bus.	Yes	Yes/Clr	0
2	<b>Post Write Non-Delivery.</b> P_SERR# is asserted because PCI 6540 was unable to deliver Posted Write data to the target before the Timeout Counter expired.	Yes	Yes/Clr	0
3	<b>Target Abort during Posted Write.</b> P_SERR# is asserted because PCI 6540 received a Target Abort when delivering Posted Write data.	Yes	Yes/Clr	0
4	<b>Master Abort during Posted Write.</b> P_SERR# is asserted because PCI 6540 received a Master Abort when delivering Posted Write data.	Yes	Yes/Clr	0
5	<b>Delayed Write Non-Delivery.</b> P_SERR# is asserted because PCI 6540 was unable to deliver Delayed Write data before the Timeout Counter expired.	Yes	Yes/Clr	0
6	<b>Delayed Read Failed.</b> P_SERR# is asserted because PCI 6540 was unable to read data from the target before the Timeout Counter expired.	Yes	Yes/Clr	0
7	<b>Delayed Transaction Master Timeout.</b> P_SERR# is asserted because a master did not repeat a Read or Write transaction before the initiator bus Master Timeout Counter expired.	Yes	Yes/Clr	0



## 6.1.2.14 Clock Run Register

Register 6-55. (CLKRUN; PCI:6Bh) Clock Run

Bit	Description	Read	Write	Value after Reset
0	<b>Secondary Clock Stop Status.</b> Values: 0 = Secondary clock not stopped 1 = Secondary clock stopped	Yes	Yes	0
1	<b>S_CLKRUN# Enable.</b> Controls the S_CLKRUN# pin. Values: 0 = Disables S_CLKRUN# pin 1 = Enables S_CLKRUN# pin	Yes	Yes	0
2	<b>Primary Clock Stop.</b> Values: 0 = Allows primary clock to stop, if secondary clock is stopped 1 = Keeps primary clock running	Yes	Yes	0
3	<b>P_CLKRUN# Enable.</b> Controls the P_CLKRUN# pin. Values: 0 = Disables P_CLKRUN# pin 1 = Enables P_CLKRUN# pin	Yes	Yes	0
4	<b>Clkrun Mode.</b> Values: 0 = Stops the secondary clock only on request from the primary bus 1 = Stops the secondary clock when the secondary bus is idle and there are no requests from the primary bus	Yes	Yes	0
7:5	<b>Reserved.</b>	Yes	No	000b

### 6.1.2.15 Private Memory Registers

Private Memory can be enabled by way of the Chip Control register (CCNTRL[2]; PCI:40h) or by using the PRV\_DEV input pin.

**Register 6-56. (PVTMBAR; PCI:6Ch) Private Memory Base**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Private Memory Base.</b> Defines the Private Memory address range Base address. The upper 12 bits corresponding to Address bits [31:20] are writable and reset to 0h. The lower 20 Address bits [19:0] are assumed to be 0h. The lower four bits are Read-Only and set to 0001b.	Yes	Yes [15:4]	1h

**Register 6-57. (PVTMLMT; PCI:6Eh) Private Memory Limit**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Private Memory Limit.</b> Defines the Private Memory address range upper limit address. The upper 12 bits corresponding to Address bits [31:20] are writable and reset to 0h. The lower 20 Address bits [19:0] are assumed to be F_FFFh. The lower four bits are Read-Only and set to 0h.	Yes	Yes [15:4]	0h

**Register 6-58. (PVTMBARU32; PCI:70h) Private Memory Base Upper 32 Bits**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Private Memory Base Upper 32 Bits.</b> Defines the upper 32-bit (bits [63:32]) Memory Base address of the Private Memory Address range.	Yes	Yes [15:4]	1h

**Register 6-59. (PVTMLMTU32; PCI:74h) Private Memory Limit Upper 32 Bits**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Private Memory Limit Upper 32 Bits.</b> Defines the upper 32-bit (bits [63:32]) Memory Limit address of the Private Memory Address range.	Yes	Yes [15:4]	0h

## 6.1.2.16 Hot Swap and Read-Only Register Control Register

Register 6-60. (HSSRRC; PCI:9Ch) Hot Swap Switch and ROR Control

Bit	Description	Read	Write	Value after Reset
0	<b>Hot Swap Extraction Switch.</b> Used to signal board extraction. If set, the board is in the inserted state. Writing 0 to this bit signals pending board extraction.	Yes	Yes	—
1	<b>Primary Port 64-Bit Extension Signals Park.</b> Value: 1 = Drives primary port PCI 64-bit extension signals P_AD[63:32], P_CBE[7:4]#, and P_PAR64 to 0	Yes	Yes	0
2	<b>Secondary Port 64-Bit Extension Signals Park.</b> Value: 1 = Drives secondary port PCI 64-bit extension signals S_AD[63:32], S_CBE[7:4]#, and S_PAR64 to 0	Yes	Yes	0
4:3	<b>Reserved.</b>	Yes	No	00b
5	<b>Downstream Translation BAR Access.</b> Value: 1 = Enables the shadowed Downstream Address Translation BARs to be accessed and written to (refer to Section 6.1.2.19)	Yes	Yes	0
6	<b>Upstream Translation BAR Access.</b> Value: 1 = Enables the shadowed Upstream Address Translation BARs to be accessed and written to (refer to Section 6.1.2.19)	Yes	Yes	0
7	<b>Read-Only Registers Write Enable.</b> Setting this bit to 1 enables writes to specific bits within these normally Read-Only registers (refer to the listed registers for further details): <ul style="list-style-type: none"> <li>• Vendor and Device IDs (PCIIDR; PCI:00h)</li> <li>• PCI Class Code (PCICCR; PCI:09h – 0Bh)</li> <li>• PCI Header Type (PCIHTR; PCI:0Eh)</li> <li>• Power Management Capabilities (PMC; PCI:DEh)</li> <li>• Power Management Control/Status (PMCSR; PCI:E0h)</li> <li>• Power Management Data (PMCDATA; PCI:E3h)</li> </ul> Bit must be cleared after the values are modified in these Read-Only registers.	Yes	Yes	0

### 6.1.2.17 GPIO[7:4], Power-Up Status, and GPIO[15:8] Registers

Register 6-61. (GPIOOD[7:4]; PCI:9Dh) GPIO[7:4] Output Data

Bit	Description	Read	Write	Value after Reset
3:0	<b>GPIO[7:4] Output Data Write 1 to Clear.</b> Writing 1 to these bits drives the corresponding signal low on the GPIO[7:4] bus, if the signal is programmed as an output. Writing 0 has no effect. Read returns the last written value.	Yes	Yes/Clr	0h
7:4	<b>GPIO[7:4] Output Data Write 1 to Set.</b> Writing 1 to these bits drives the corresponding signal high on the GPIO[7:4] bus, if the signal is programmed as an output. Writing 0 has no effect. Read returns the last written value.	Yes	Yes/Set High	0h

Register 6-62. (GPIOOE[7:4]; PCI:9Eh) GPIO[7:4] Output Enable

Bit	Description	Read	Write	Value after Reset
3:0	<b>GPIO[7:4] Output Enable Write 1 to Clear.</b> Writing 1 to these bits configures the corresponding signal on the GPIO[7:4] bus as an input. Writing 0 has no effect. Read returns the last written value.	Yes	Yes/Clr	0h
7:4	<b>GPIO[7:4] Output Enable Write 1 to Set.</b> Writing 1 to these bits configures the corresponding signal on the GPIO[7:4] bus as an output. Writing 0 has no effect. Read returns the last written value.	Yes	Yes/Set High	0h

Register 6-63. (GPIOID[7:4]; PCI:9Fh) GPIO[7:4] Input Data

Bit	Description	Read	Write	Value after Reset
3:0	<b>Reserved.</b>	Yes	No	0h
7:4	<b>GPIO[7:4] Input Data.</b> Reads the GPIO[7:4] pin state. The state is updated on the PCI Clock cycle following a change in GPIO[7:4] state.	Yes	No	—

**Register 6-64. (PWRUPSR; PCI:A0h) Power-Up Status**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Power-Up Latched Status Bits.</b> Upon PWRGD (power good), the GPIO[15:8] status is latched into PWRUPSR. Select pin status for desired option setting or checking. Recommended use: <ul style="list-style-type: none"> <li>• <b>GPIO15—Primary Power State.</b> Value of 1h indicates primary port power is stable.</li> <li>• <b>GPIO14—Secondary Power State.</b> Value of 1h indicates secondary port power is stable.</li> <li>• <b>GPIO13—100 MHz PCI-X.</b> Value of 1h indicates secondary PCI-X Bus is running at 100 MHz.</li> </ul>	Yes	No	GPIO[15:8]

**Register 6-65. (GPIOOD[15:8]; PCI:A1h) GPIO[15:8] Output Data**

Bit	Description	Read	Write	Value after Reset
7:0	<b>GPIO[15:8] Output Data.</b> Values written to this register are output on the GPIO[15:8] pins, if enabled. Values: 0h = Low 1h = High	Yes	Yes	0h

**Register 6-66. (GPIOOE[15:8]; PCI:A2h) GPIO[15:8] Output Enable**

Bit	Description	Read	Write	Value after Reset
7:0	<b>GPIO[15:8] Output Enable.</b> Writing 1 to these bits configures the corresponding signal on the GPIO[15:8] bus as an output. Writing 0 configures the corresponding signal as an input.	Yes	Yes	0h

**Register 6-67. (GPIOID[15:8]; PCI:A3h) GPIO[15:8] Input Data**

Bit	Description	Read	Write	Value after Reset
7:0	<b>GPIO[15:8] Input Data.</b> Reads the GPIO[15:8] pin state. The state is updated on the PCI Clock cycle following a change in GPIO[15:8] state.	Yes	No	—

### 6.1.2.18 Sticky Scratch and Extended Registers

There are eight, 32-bit Sticky Scratch registers available in the PCI 6540 (SCRATCHx; EXT:00h to 07h). Address translation registers are also located in the Extended Register Area. The extended registers

are accessed by way of the Extended Register Index and Data registers (EXTRIDX; PCI:D3h and EXTRDATA; PCI:D4h, respectively).

The Base Address registers at index 08h to 0Fh are discussed in Section 6.1.2.19.

**Table 6-2. Extended Register Map—Offset from Extended Register Index, Transparent Mode**

Extended Register Index	31	24	23	16	15	8	7	0	Writable	Serial EEPROM Writable
00h	32-Bit Sticky 0								Yes	No
01h	32-Bit Sticky 1								Yes	No
02h	32-Bit Sticky 2								Yes	No
03h	32-Bit Sticky 3								Yes	No
04h	32-Bit Sticky 4								Yes	No
05h	32-Bit Sticky 5								Yes	No
06h	32-Bit Sticky 6								Yes	No
07h	32-Bit Sticky 7								Yes	No

**Notes:** When the serial EEPROM is set to initialize for Universal Non-Transparent mode applications, these registers also activate translation in Universal Transparent mode if PRV\_DEV=1. (Refer to Section 7, "Serial EEPROM.")

Refer to the individual register descriptions to determine which bits are writable.

**Register 6-68. (SCRATCHx; Ext:00h – 07h) 32-Bit Sticky Scratch**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Sticky Scratch.</b> Upon Power Good, the values of these registers are undefined. After Power is Good, P_RSTIN# and/or S_RSTIN# assertion does not affect their current value.	Yes	Yes	—

**Register 6-69. (EXTRIDX; PCI:D3h) Extended Register Index**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Extended Index Address.</b> Index address for Extended registers.	Yes	Yes	—

**Register 6-70. (EXTRDATA; PCI:D4h) Extended Register Data**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Extended Register Data.</b> Configuration Write causes the data presented at this port to be written into the register addressed by the Extended Register Index (EXTRIDX; PCI:D3h). Configuration Read causes the data from the register addressed by the Extended Register Index to be placed into and read from EXTRDATA.	Yes	Yes	—

### 6.1.2.19 Address Translation Control Registers

When using the PCI 6540 in standard Transparent mode, the Address Translation Control registers should remain in the default state. These registers are used only when address translation is required in Transparent mode. (Refer to Section 10.7.2, “Transparent Mode Address Translation,” for further details.)

The Address Translation Enable Control bits enable or disable only the Address Translation functions. These bits do not control whether the Memory window is open. Take care to ensure that there is a valid Memory window in the Memory map of the host on that port.

The Base Address registers at extended register index 08h to 0Fh (refer to Table 6-4 on page 6-45) are accessible only by reading/writing through the Extended Register Index and Data registers.

**Table 6-3. PCI Configuration Shadowed Registers (Used in Transparent Address Translation)**

PCI Configuration Register Address Primary Offset	To ensure software compatibility with other versions of the PCI 6540 family and to ensure compatibility with future enhancements, write 0 to all unused bits.	PCI Writable	Serial EEPROM Writable
	31 0		
10h	Downstream I/O BAR 0	Only if HSSRRC[5]=1; PCI:9Ch	No
14h	Downstream Memory BAR 1	Only if HSSRRC[5]=1; PCI:9Ch	No
18h	Downstream Memory BAR 2 or Downstream Memory BAR 1 Upper 32 Bits	Only if HSSRRC[5]=1; PCI:9Ch	No
10h	Upstream I/O or Memory BAR 0	Only if HSSRRC[6]=1; PCI:9Ch	No
14h	Upstream Memory BAR 1	Only if HSSRRC[6]=1; PCI:9Ch	No
18h	Upstream Memory BAR 2 or Upstream Memory BAR 1 Upper 32 Bits	Only if HSSRRC[6]=1; PCI:9Ch	No

**Notes:** The registers listed in Table 6-3 can be written to only when the appropriate Hot Swap Switch and Read-Only register bit is set to 1 (HSSRRC[6 or 5]=1; PCI:9Ch). To allow normal device operation, HSSRRC[6:5] must be reset to 00b after the registers are configured.

Refer to the individual register descriptions to determine which bits are writable.



**Register 6-71. (PCIBAR0; Primary PCI:10h) PCI Downstream I/O BAR 0**

Bit	Description	Read	Write	Value after Reset
0	<b>Memory Space Indicator.</b> Writing 0 indicates the register maps into Memory space. Writing 1 indicates that the register maps into I/O space. <i>Must be set to 1 for correct operation.</i>	Yes	Only if HSSRRC[5]=1; PCI:9Ch	0
1	<b>Reserved. Must be set to 0.</b>	Yes	Only if HSSRRC[5]=1; PCI:9Ch	0
31:2	<b>Base Address.</b> Base address for downstream accesses.	Yes	Only if HSSRRC[5]=1; PCI:9Ch	0h

**Register 6-72. (PCIBAR1; Primary PCI:14h) PCI Downstream Memory BAR 1**

Bit	Description	Read	Write	Value after Reset
0	<b>Memory Space Indicator.</b> Writing 0 indicates the register maps into Memory space.	Yes	No	0
2:1	<b>Register Location.</b> Values: 00b = Locate anywhere in 32-bit Memory Address space 01b = PCI r2.1, Locate below 1-MB Memory Address space PCI r2.3, <b>Reserved</b>	Yes	Only if HSSRRC[5]=1; PCI:9Ch	00b
3	<b>Prefetchable.</b> Writing 1 indicates that there are no side effects on reads.	Yes	Only if HSSRRC[5]=1; PCI:9Ch	0
31:4	<b>Base Address.</b> Base Address for downstream accesses.	Yes	Only if HSSRRC[5]=1; PCI:9Ch	0h

**Register 6-73. (PCIBAR2; Primary PCI:18h) PCI Downstream Memory BAR 2 or Downstream Memory BAR 1 Upper 32 Bits**

Bit	Description	Read	Write	Value after Reset
0	<b>Memory Space Indicator.</b> Writing 0 indicates the register maps into Memory space.	Yes	No	0
2:1	<b>Register Location.</b> Values: 00b = Locate anywhere in 32-bit Memory Address space 01b = PCI r2.1, Locate below 1-MB Memory Address space PCI r2.3, <b>Reserved</b>	Yes	Only if HSSRRC[5]=1; PCI:9Ch	00b
3	<b>Prefetchable.</b> Writing 1 indicates that there are no side effects on reads.	Yes	Only if HSSRRC[5]=1; PCI:9Ch	0
31:4	<b>Base Address.</b> Base address for downstream accesses.	Yes	Only if HSSRRC[5]=1; PCI:9Ch	0h

**Note:** When using 64-bit addressing for PCIBAR1, PCIBAR2 becomes the upper 32 bits of the PCIBAR1 address. (Refer to DWNBAR1MSK[14].)

**Register 6-74. (PCIUBAR0; Primary PCI:10h) PCI Upstream I/O or Memory BAR 0**

Bit	Description	Read	Write	Value after Reset
0	<b>Memory Space Indicator.</b> Writing 0 indicates the register maps into Memory space. Writing 1 indicates that the register maps into I/O space.	Yes	Only if HSSRRC[6]=1; PCI:9Ch)	0
2:1	<b>Register Location (If Memory Space).</b> When mapped into I/O space (PCIUBAR0[0]=1), bit 1 is always 0 and bit 2 is included in the Base address (PCIBAR0[31:4]). Values: 00b = Locate anywhere in 32-bit Memory Address space 01b = <i>PCI r2.1</i> , Locate below 1-MB Memory Address space <i>PCI r2.3, Reserved</i>	Yes	Only if HSSRRC[6]=1; PCI:9Ch)	00b
3	<b>Prefetchable (If Memory Space).</b> Writing 1 indicates that there are no side effects on reads. When mapped into I/O space (PCIUBAR0[0]=1), bit 3 is included in the Base address (PCIUBAR0[31:4]).	Yes	Only if HSSRRC[6]=1; PCI:9Ch)	0
31:4	<b>Base Address.</b> Base address for upstream accesses.	Yes	Only if HSSRRC[6]=1; PCI:9Ch)	0h

**Register 6-75. (PCIUBAR1; Primary PCI:14h) PCI Upstream Memory BAR 1**

Bit	Description	Read	Write	Value after Reset
0	<b>Memory Space Indicator.</b> Writing 0 indicates the register maps into Memory space.	Yes	No	0
2:1	<b>Register Location.</b> Values: 00b = Locate anywhere in 32-bit Memory Address space 01b = <i>PCI r2.1</i> , Locate below 1-MB Memory Address space <i>PCI r2.3, Reserved</i>	Yes	Only if HSSRRC[6]=1; PCI:9Ch)	00b
3	<b>Prefetchable.</b> Writing 1 indicates that there are no side effects on reads.	Yes	Only if HSSRRC[6]=1; PCI:9Ch)	0
31:4	<b>Base Address.</b> Base address for upstream accesses.	Yes	Only if HSSRRC[6]=1; PCI:9Ch)	0h

**Register 6-76. (PCIUBAR2; Primary PCI:18h) PCI Upstream Memory BAR 2 or Upstream Memory Bar 1 Upper 32 Bits**

Bit	Description	Read	Write	Value after Reset
0	<b>Memory Space Indicator.</b> Writing 0 indicates the register maps into Memory space.	Yes	No	0
2:1	<b>Register Location.</b> Values: 00b = Locate anywhere in 32-bit Memory Address space 01b = PCI r2.1, Locate below 1-MB Memory Address space PCI r2.3, <b>Reserved</b>	Yes	Only if HSSRRC[6]=1; PCI:9Ch)	00b
3	<b>Prefetchable.</b> Writing 1 indicates that there are no side effects on reads.	Yes	Only if HSSRRC[6]=1; PCI:9Ch)	0
31:4	<b>Base Address.</b> Base address for upstream accesses.	Yes	Only if HSSRRC[6]=1; PCI:9Ch)	0h

**Note:** When using 64-bit addressing for PCIUBAR1, PCIUBAR2 becomes the upper 32 bits of the PCIUBAR1 address. (Refer to UPSBAR1MSK[14].)

**Table 6-4. Extended Register Map (Used in Transparent Address Translation)—Offset from Extended Register Index**

Extended Register Index	3124231615870								Writable	Serial EEPROM Writable
08h	Upstream BAR 0 Translation Address								Yes	Yes
09h	Upstream BAR 1 Translation Address								Yes	Yes
0Ah	Upstream BAR 2 or Upstream BAR 1 Upper 32 Bits Translation Address								Yes	Yes
0Bh	Upstream Translation Enable	Upstream BAR 2 Translation Mask		Upstream BAR 1 Translation Mask		Upstream BAR 0 Translation Mask		Yes	Yes	
0Ch	Downstream BAR 0 Translation Address								Yes	Yes
0Dh	Downstream BAR 1 Translation Address								Yes	Yes
0Eh	Downstream BAR 2 or Downstream BAR 1 Upper 32 Bits Translation Address								Yes	Yes
0Fh	Downstream Translation Enable	Downstream BAR 2 Translation Mask		Downstream BAR 1 Translation Mask		Downstream BAR 0 Translation Mask		Yes	Yes	

**Notes:** When the serial EEPROM is set to initialize for Universal Non-Transparent mode applications, these registers also activate translation in Universal Transparent mode if PRV\_DEV=1. (Refer to Section 10.7.2, “Transparent Mode Address Translation,” on page 10-6 and Section 10.7.3, “Non-Transparent Mode Address Translation,” on page 10-11 for further details.)

Refer to the individual register descriptions to determine which bits are writable.

**Register 6-77. (UPSTNBAR0; Ext:08h) Upstream BAR 0 Translation Address**

Bit	Description	Read	Write	Value after Reset
31:0	<p><b>Upstream BAR 0 Translation Address.</b> Bits [11:0] are Read-Only and always 0. Only Address bits [31:12] are translated. Lower Address bits are passed.</p> <p><b>Note:</b> Translation address must align with window-size boundary.</p>	Yes	Yes [31:12]; Serial EEPROM	0h

**Register 6-78. (UPSTNBAR1; Ext:09h) Upstream BAR 1 Translation Address**

Bit	Description	Read	Write	Value after Reset
31:0	<p><b>Upstream BAR 1 Translation Address.</b> Bits [19:0] are Read-Only and always 0. Only Address bits [31:20] are translated. Lower Address bits are passed.</p> <p><b>Note:</b> Translation address must align with window-size boundary.</p>	Yes	Yes [31:20]; Serial EEPROM	0h

**Register 6-79. (UPSTNBAR2; Ext:0Ah) Upstream BAR 2 Translation Address or Upstream BAR 1 Upper 32 Bits**

Bit	Description	Read	Write	Value after Reset
31:0	<p><b>Upstream BAR 2 Translation Address or Upstream BAR 1 Upper 32 Bits.</b> Bits [11:0] are Read-Only and always 0. Only Address bits [31:12] are translated. Lower Address bits are passed.</p> <p>If PCIUBAR1 is configured as a 64-bit BAR (UPSBAR1MSK[14]=1; EXT:0Bh), UPSTNBAR2 contains the upper 32 bits of the BAR 1 Translation address.</p> <p><b>Note:</b> Translation address must align with window-size boundary.</p>	Yes	Yes [31:12]; Serial EEPROM	0h

**Register 6-80. (UPSBAR0MSK; Ext:0Bh) Upstream BAR 0 Translation Mask**

Bit	Description	Read	Write	Value after Reset
4:0	<b>Address Mask MSB Position.</b> Number of Local Address bits for BAR 0 mask.	Yes	Yes; Serial EEPROM	1Fh
5	<b>Reserved.</b>	Yes	No	0
6	<b>BAR Type.</b> Values: 0 = BAR 0 points to I/O space 1 = BAR 0 points to Memory space	Yes	Yes; Serial EEPROM	0
7	<b>Prefetchable.</b> Values: 0 = Region pointed to by BAR 0 is not prefetchable 1 = Region pointed to by BAR 0 is in a Prefetchable Memory region	Yes	Yes; Serial EEPROM	0

**Register 6-81. (UPSBAR1MSK; Ext:0Bh) Upstream BAR 1 Translation Mask**

Bit	Description	Read	Write	Value after Reset
13:8	<b>Address Mask MSB Position.</b> Number of Local Address bits for BAR 1 mask.	Yes	Yes; Serial EEPROM	3Fh
14	<b>BAR Type.</b> Values: 0 = BAR 1 is a 32-bit BAR 1 = BAR 1 is a 64-bit BAR	Yes	Yes; Serial EEPROM	0
15	<b>Prefetchable.</b> Values: 0 = Region pointed to by BAR 1 is not prefetchable 1 = Region pointed to by BAR 1 is in a Prefetchable Memory region	Yes	Yes; Serial EEPROM	0

**Register 6-82. (UPSBAR2MSK; Ext:0Bh) Upstream BAR 2 Translation Mask**

Bit	Description	Read	Write	Value after Reset
20:16	<b>Address Mask MSB Position.</b> Number of Local Address bits for BAR 2 mask.	Yes	Yes; Serial EEPROM	1Fh
22:21	<b>Reserved.</b>	Yes	No	00b
23	<b>Prefetchable.</b> Values: 0 = Region pointed to by BAR 2 is not prefetchable 1 = Region pointed to by BAR 2 is in a Prefetchable Memory region	Yes	Yes; Serial EEPROM	0

Register 6-83. (UPSTNE; Ext:0Bh) Upstream Translation Enable

Bit	Description	Read	Write	Value after Reset
24	<b>Upstream BAR 0 Enable.</b> If 1, address translation using BAR 0 is enabled.	Yes	Yes; Serial EEPROM	0
25	<b>Upstream BAR 1 Enable.</b> If 1, address translation using BAR 1 is enabled.	Yes	Yes; Serial EEPROM	0
26	<b>Upstream BAR 2 Enable.</b> If 1, address translation using BAR 2 is enabled.	Yes	Yes; Serial EEPROM	0
31:27	<i>Reserved.</i>	Yes	No	0h

Register 6-84. (DWNTNBAR0; Ext:0Ch) Downstream BAR 0 Translation Address

Bit	Description	Read	Write	Value after Reset
31:0	<b>Downstream BAR 0 Translation Address.</b> Bits [11:0] are Read-Only and always 0. Only Address bits [31:12] are translated. Lower Address bits are passed. <i>Note:</i> Translation address must align with window-size boundary.	Yes	Yes [31:12]; Serial EEPROM	0h

Register 6-85. (DWNTNBAR1; Ext:0Dh) Downstream BAR 1 Translation Address

Bit	Description	Read	Write	Value after Reset
31:0	<b>Downstream BAR 1 Translation Address.</b> Bits [19:0] are Read-Only and always 0. Only Address bits [31:20] are translated. Lower Address bits are passed. <i>Note:</i> Translation address must align with window-size boundary.	Yes	Yes [31:20]; Serial EEPROM	0h

Register 6-86. (DWNTNBAR2; Ext:0Eh) Downstream BAR 2 or  
Downstream Memory BAR 1 Upper 32 Bits Translation Address

Bit	Description	Read	Write	Value after Reset
31:0	<b>Downstream BAR 2 or Downstream Memory BAR 1 Upper 32 Bits Translation Address.</b> Bits [11:0] are Read-Only and always 0. Only Address bits [31:12] are translated. Lower Address bits are passed. If PCIBAR1 is configured as a 64-bit BAR (DWNBAR1MSK[14]=1; Ext:0Fh), DWNTNBAR2 contains the upper 32 bits of the BAR 1 Translation address. <i>Note:</i> Translation address must align with window-size boundary.	Yes	Yes [31:12]; Serial EEPROM	0h

**Register 6-87. (DWNBAR0MSK; Ext:0Fh) Downstream BAR 0 Translation Mask**

Bit	Description	Read	Write	Value after Reset
4:0	<b>Address Mask MSB Position.</b> Number of Local Address bits for BAR 0 mask.	Yes	Yes; Serial EEPROM	1Fh
5	<b>Reserved.</b>	Yes	No	0
6	<b>BAR Type. <i>Must be set to 1.</i></b> Values: 0 = BAR 0 points to Memory space 1 = BAR 0 points to I/O space	Yes	Yes; Serial EEPROM	0
7	<b>Reserved. <i>Must be set to 0.</i></b> This bit is normally used to determine whether the region pointed to by the BAR is prefetchable. However, this BAR must map into I/O space, which by definition is <i>not</i> prefetchable. Therefore, this bit must be set to 0.  <b>Note:</b> <i>In Non-Transparent mode, this BAR can map into either Memory or I/O space, which is why the bit is writable.</i>	Yes	Yes; Serial EEPROM	0

**Register 6-88. (DWNBAR1MSK; Ext:0Fh) Downstream BAR 1 Translation Mask**

Bit	Description	Read	Write	Value after Reset
13:8	<b>Address Mask MSB Position.</b> Number of Local Address bits for BAR 1 mask.	Yes	Yes; Serial EEPROM	3Fh
14	<b>BAR Type.</b> Values: 0 = BAR 1 is a 32-bit BAR 1 = BAR 1 is a 64-bit BAR	Yes	Yes; Serial EEPROM	0
15	<b>Prefetchable.</b> Values: 0 = Region pointed to by BAR 1 is not prefetchable 1 = Region pointed to by BAR 1 is in a Prefetchable Memory region	Yes	Yes; Serial EEPROM	0

**Register 6-89. (DWNBAR2MSK; Ext:0Fh) Downstream BAR 2 Translation Mask**

Bit	Description	Read	Write	Value after Reset
20:16	<b>Address Mask MSB Position.</b> Number of Local Address bits for BAR 2 mask.	Yes	Yes; Serial EEPROM	1Fh
22:21	<b>Reserved.</b>	Yes	No	00b
23	<b>Prefetchable.</b> Values: 0 = Region pointed to by BAR 2 is not prefetchable 1 = Region pointed to by BAR 2 is in a Prefetchable Memory region	Yes	Yes; Serial EEPROM	0

**Register 6-90. (DWNTNE; Ext:0Fh) Downstream Translation Enable**

Bit	Description	Read	Write	Value after Reset
24	<b>Downstream BAR 0 Enable.</b> If 1, address translation using BAR 0 is enabled.	Yes	Yes; Serial EEPROM	0
25	<b>Downstream BAR 1 Enable.</b> If 1, address translation using BAR 1 is enabled.	Yes	Yes; Serial EEPROM	0
26	<b>Downstream BAR 2 Enable.</b> If 1, address translation using BAR 2 is enabled.	Yes	Yes; Serial EEPROM	0
31:27	<b>Reserved.</b>	Yes	No	0h



### 6.1.2.20 Power Management Capability Registers

The PMC[12:9, 1:0]; PCI:DEh, PMCDATA; PCI:E3h, and PMCSR; PCI:E0h Power Management registers are normally Read-Only. However, their default values can be changed by firmware or software by setting the Read-Only Registers Write Enable bit (HSSRRC[7]=1;

PCI:9Ch). After modifying these registers, the Write Enable bit must be cleared to preserve the Read-Only nature of these registers. It should be noted that the HSSRRC[7] state does *not* affect Write accesses to PMCSR[15, 8].

#### Register 6-91. (PMCAPID; PCI:DCh) Power Management Capability ID

Bit	Description	Read	Write	Value after Reset
7:0	<b>Power Management Capability ID.</b> PCI-SIG-issued Capability ID for Power Management is 1h.	Yes	No	1h

#### Register 6-92. (PMNEXT; PCI:DDh) Power Management Next Capability Pointer

Bit	Description	Read	Write	Value after Reset
7:0	<b>Next_Cap Pointer.</b> Provides an offset into PCI Configuration space for the Hot Swap capability location in the New Capabilities Linked List.	Yes	No	E4h

Register 6-93. (PMC; PCI:DEh) Power Management Capabilities

Bit	Description	Read	Write	Value after Reset
2:0	<b>Version.</b> Set to 001b, indicates that this function complies with <i>PCI Power Mgmt. r1.1</i> .	Yes	Only if HSSRRRC[7]=1; Serial EEPROM	001b
3	<b>PME Clock.</b> Because PCI 6540 does not require the PCI clock for PME#, set this bit to 0.	Yes	Only if HSSRRRC[7]=1; Serial EEPROM	0
4	<b>Auxiliary Power Source.</b> Because PCI 6540 does not support PME# while in a D <sub>3cold</sub> state, this bit is always set to 0.	Yes	Only if HSSRRRC[7]=1; Serial EEPROM	0
5	<b>Device-Specific Initialization (DSI).</b> Returns 0, indicating PCI 6540 does not require special initialization.	Yes	Only if HSSRRRC[7]=1; Serial EEPROM	0
8:6	<b>Reserved.</b>	Yes	No	000b
9	<b>D<sub>1</sub> Support.</b> Returns 1, indicating that PCI 6540 supports the D <sub>1</sub> device power state.	Yes	Only if HSSRRRC[7]=1; Serial EEPROM	1
10	<b>D<sub>2</sub> Support.</b> Returns 1, indicating that PCI 6540 supports the D <sub>2</sub> device power state.	Yes	Only if HSSRRRC[7]=1; Serial EEPROM	1
15:11	<b>PME Support.</b> Values: XXXX1b = PME# can be asserted from D <sub>0</sub> XXX1Xb = PME# can be asserted from D <sub>1</sub> XX1XXb = PME# can be asserted from D <sub>2</sub> X1XXXb = PME# can be asserted from D <sub>3hot</sub> 1XXXXb = PME# can be asserted from D <sub>3cold</sub>	Yes	Only if HSSRRRC[7]=1; Serial EEPROM	01111b

**Register 6-94. (PMCSR; PCI:E0h) Power Management Control/Status**

Bit	Description	Read	Write	Value after Reset
1:0	<b>Power State.</b> Used to determine the current power state of a function and to set the function into a new power state. Values: 00b = D <sub>0</sub> 01b = D <sub>1</sub> 10b = D <sub>2</sub> 11b = D <sub>3hot</sub>	Yes	Yes; Serial EEPROM	00b
7:2	<b>Reserved.</b>	Yes	No	0h
8	<b>PME Enable.</b> Enables the PME# output pin. Values: 0 = PME# output disabled 1 = PME# output enabled  <b>Note:</b> In Transparent mode, P_PME# and S_PME# should be pulled high and not used. PME# output is always P_PME#. In Non-Transparent mode, the PME# output is either P_PME# or S_PME#, depending on the P_BOOT value.	Yes	Yes; Serial EEPROM	0
12:9	<b>Data Select.</b> Returns 0h, indicating PCI 6540 does not return dynamic data.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0h
14:13	<b>Data Scale.</b> Returns 00b when read. PCI 6540 does not return dynamic data.	Yes	No	00b
15	<b>PME Status.</b> Set to 0, because PCI 6540 does not support PME# signaling.	Yes	Yes; Serial EEPROM	0

**Register 6-95. (PMCSR\_BSE; PCI:E2h) PMCSR Bridge Supports Extensions**

Bit	Description	Read	Write	Value after Reset
5:0	<b>Reserved.</b>	Yes	No	0h
6	<b>B<sub>2</sub>/B<sub>3</sub> Support for D<sub>3hot</sub>.</b> Reflects BPCC_EN input pin state. Value of 1 indicates that when PCI 6540 is programmed to D <sub>3hot</sub> state, the secondary bus clock is stopped.	Yes	No	—
7	<b>Bus Power Control Enable.</b> Reflects BPCC_EN input pin state. Value of 1 indicates that the secondary bus Power Management state follows that of PCI 6540, with one exception—D <sub>3hot</sub> .	Yes	No	—

**Register 6-96. (PMCDATA; PCI:E3h) Power Management Data**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Power Management Data.</b> Serial EEPROM or ROR Write controlled loadable, but is Read-Only during normal operation.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0h

### 6.1.2.21 Hot Swap Capability Registers

**Register 6-97. (HS\_CNTL; PCI:E4h) Hot Swap Control**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Hot Swap Capability ID.</b> PCI-SIG-issued Capability ID for Hot Swap is 06h.	Yes	No	06h

**Register 6-98. (HS\_NEXT; PCI:E5h) Hot Swap Next Capability Pointer**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Next_Cap Pointer.</b> Provides an offset into PCI Configuration space for the VPD capability location in the New Capabilities Linked List.	Yes	No	E8h

**Register 6-99. (HS\_CSR; PCI:E6h) Hot Swap Control/Status**

Bit	Description	Read	Write	Value after Reset
0	<b>Device Hiding Arm (DHA).</b> DHA is set to 1 by hardware when the Hot Swap port PCI RSTIN# becomes inactive and the handle switch remains unlocked. Handle locking clears this bit. Values: 0 = Disarm Device Hiding 1 = Arm Device Hiding	Yes	Yes	0
1	<b>ENUM# Mask Status (EIM).</b> Enables or disables ENUM# assertion. Values: 0 = Enables ENUM# assertion 1 = Masks ENUM# assertion	Yes	Yes	0
2	<b>Pending INSERT or EXTRACT (PIE).</b> Set when INS or EXT is 1 or INS is armed (write 1 to EXT bit). Values: 0 = Neither is pending 1 = Either an insertion or an extraction is in progress	Yes	No	—
3	<b>LED Status (LOO).</b> Indicates whether LED is ON or OFF. Values: 0 = LED OFF 1 = LED ON	Yes	Yes	0
5:4	<b>Programming Interface (PI).</b> Hardcoded at 01b—INS, EST, LOO, EIM, PIE, and Device Hiding supported.	Yes	No	01b
6	<b>Extraction State (EXT).</b> Set by hardware when the ejector handle is unlocked and INS=0.	Yes	Yes/Clr	—
7	<b>Insertion State (INS).</b> Set by hardware when the Hot Swap port RSTIN# is de-asserted, serial EEPROM autoload is completed, and ejector handle is locked. Writing 1 to EXT bit also arms INS.	Yes	Yes/Clr	—
15:8	<b>Reserved.</b>	Yes	No	0h

## 6.1.2.22 VPD Capability Registers

Register 6-100. (PVPDID; PCI:E8h) Vital Product Data Capability ID

Bit	Description	Read	Write	Value after Reset
7:0	<b>Vital Product Data Capability ID.</b> PCI-SIG-issued Capability ID for VPD is 03h.	Yes	No	03h

Register 6-101. (PVPD\_NEXT; PCI:E9h) Vital Product Data Next Capability Pointer

Bit	Description	Read	Write	Value after Reset
7:0	<b>Next_Cap Pointer.</b> Provides offset into PCI Configuration space for the PCI-X capability location in the New Capabilities Linked List.	Yes	No	F0h

Register 6-102. (PVPDAD; PCI:EAh) Vital Product Data Address

Bit	Description	Read	Write	Value after Reset
1:0	<b>Reserved.</b>	Yes	No	00b
7:2	<b>VPD Address.</b> Offset into the serial EEPROM to location where data is written and read. PCI 6540 accesses the serial EEPROM at address PVPDAD[7:2]+40h. The 40h offset ensures that VPD accesses do not overwrite the PCI 6540 serial EEPROM Configuration data stored in serial EEPROM locations 00h to 3Fh.	Yes	Yes	0
14:8	<b>Reserved.</b>	Yes	No	0h
15	<b>VPD Operation.</b> Writing 0 generates a Read cycle from the serial EEPROM at the VPD address specified in PVPDAD[7:2]. This bit remains at logic 0 until the serial EEPROM cycle is complete, at which time the bit is set to 1. Data for reads is available in the VPD Data register (PVPDATA; PCI:ECh). Writing 1 generates a Write cycle to the serial EEPROM at the VPD address specified in PVPDAD[7:2]. Remains at logic 1, until the serial EEPROM cycle is completed, at which time the bit is cleared to 0. Place data for writes into the VPD Data register.	Yes	Yes	0

Register 6-103. (PVPDATA; PCI:ECh) VPD Data

Bit	Description	Read	Write	Value after Reset
31:0	<b>VPD Data (Serial EEPROM Data).</b> The least significant byte of this register corresponds to the byte of VPD at the address specified by the VPD Address register (PVPDAD[7:2]; PCI:EAh). Data is read from or written to PVPDATA, using standard Configuration accesses.	Yes	Yes	0h

### 6.1.2.23 PCI-X Capability Registers

**Register 6-104. (PCIXCAPID; PCI:F0h) PCI-X Capability ID**

Bit	Description	Read	Write	Value after Reset
7:0	<b>PCI-X Capability ID.</b> PCI-SIG-issued Capability ID for PCI-X is 07h.	Yes	No	07h

**Register 6-105. (PCIX\_NEXT; PCI:F1h) PCI-X Next Capability Pointer**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Next_Cap Pointer.</b> Provides an offset into PCI Configuration space for the location of the next capability in the New Capabilities Linked List. Set to 0h in Transparent mode to indicate the end of the Capabilities list.	Yes	No	0h

**Register 6-106. (PCIXSSR; PCI:F2h) PCI-X Secondary Status**

Bit	Description	Read	Write	Value after Reset
0	<b>64-Bit Device.</b> Indicates the PCI 6540 secondary AD interface width. Reflects result of the secondary Reset time inquiry, to determine 64-bit device existence. Values: 0 = 32-bit bus data width 1 = 64-bit bus data width	Yes	No	1
1	<b>133 MHz Capable.</b> Indicates PCI 6540 secondary interface is capable of 133 MHz operations in PCI-X mode. Values: 0 = Maximum operating frequency is 66 MHz 1 = Maximum operating frequency is 133 MHz	Yes	No	—
2	<b>Split Completion Discarded.</b> Set if PCI 6540 discards a Split Completion moving toward the secondary bus, because requester would not accept it. Values: 0 = Split Completion not discarded 1 = Split Completion discarded	Yes	Yes/Clr	0
3	<b>Unexpected Split Completion.</b> Set if an Unexpected Split Completion with a Requester ID equal to PCI 6540 secondary Bus Number, Device Number 00h, and Function Number 000b is received on PCI 6540 secondary interface. Values: 0 = No Unexpected Split Completion received 1 = Unexpected Split Completion received	Yes	Yes/Clr	0

Register 6-106. (PCIXSSR; PCI:F2h) PCI-X Secondary Status (Continued)

Bit	Description	Read	Write	Value after Reset																					
4	<b>Split Completion Overrun.</b> Set if PCI 6540 terminates a Split Completion on the secondary bus, with Retry or Disconnect at the next ADB, because PCI 6540 buffers are full. Values: 0 = Bridge accepted all Split Completions 1= Bridge terminated a Split Completion, with a Retry or Disconnect at the next ADB because the bridge buffers are full	Yes	Yes/Clr	0																					
5	<b>Split Request Delayed.</b> Set if PCI 6540 received a request to forward a transaction on the secondary bus, but cannot because there is insufficient space within the limit specified in the PCI-X Downstream Split Transaction register Split Transaction Commitment Limit bits (PCIXDNSTR[31:16]; PCI:FCh). Values: 0 = Bridge did not delay a Split Request 1 = Bridge delayed a Split Request	Yes	Yes/Clr	0																					
8:6	<b>Secondary Clock Frequency.</b> Enables configuration software to determine to which mode (and in PCI-X mode, to which frequency) the PCI 6540 set the secondary bus the last time S_RSTIN# was asserted. Values: <table><tr><th>Bits</th><th>Maximum Clock Frequency (MHz)</th><th>Minimum Clock Period (ns)</th></tr><tr><td>[8:6]</td><td></td><td></td></tr><tr><td>000b</td><td>Conventional PCI mode</td><td>N/A</td></tr><tr><td>001b</td><td>PCI-X 66</td><td>15</td></tr><tr><td>010b</td><td>PCI-X 100</td><td>10</td></tr><tr><td>011b</td><td>PCI-X 133</td><td>7.5</td></tr><tr><td>1XXb</td><td><b>Reserved</b></td><td><b>Reserved</b></td></tr></table>	Bits	Maximum Clock Frequency (MHz)	Minimum Clock Period (ns)	[8:6]			000b	Conventional PCI mode	N/A	001b	PCI-X 66	15	010b	PCI-X 100	10	011b	PCI-X 133	7.5	1XXb	<b>Reserved</b>	<b>Reserved</b>	Yes	No	—
Bits	Maximum Clock Frequency (MHz)	Minimum Clock Period (ns)																							
[8:6]																									
000b	Conventional PCI mode	N/A																							
001b	PCI-X 66	15																							
010b	PCI-X 100	10																							
011b	PCI-X 133	7.5																							
1XXb	<b>Reserved</b>	<b>Reserved</b>																							
15:9	<b>Reserved.</b>	Yes	No	0h																					

Register 6-107. (PCIXBSR; PCI:F4h) PCI-X Bridge Status

Bit	Description	Read	Write	Value after Reset
2:0	<b>Function Number.</b> Indicates the number of this function—the number in AD[10:8] of a Type 0 Configuration Transaction address to which this bridge responds. The function uses this number as part of its Requester and Completer IDs (set to 000b). PCI 6540 uses the Bus, Device, and Function Numbers fields to create the Completer ID when responding with a Split Completion to a read of an internal bridge register.	Yes	No	000b
7:3	<b>Device Number.</b> Indicates Device Number (the number in AD[15:11] of the address of a Type 0 Configuration transaction) assigned to PCI 6540.	Yes	No	11111b
15:8	<b>Bus Number.</b> Additional addresses from which the Primary Bus Number register contents (in the Type 01h Configuration Space header) are read.	Yes	No	—
16	<b>64-Bit Device.</b> Indicates the bridge AD bus data width. This bit is the inverse of the DEV64# input. Values: 0 = 32-bit bus data width 1 = 64-bit bus data width	Yes	No	Inverse of DEV64# Input
17	<b>133 MHz Capability.</b> Indicates bridge primary interface is capable of 133 MHz operations in PCI-X mode. Values: 0 = Device maximum frequency is 66 MHz 1 = Device maximum frequency is 133 MHz	Yes	No	1
18	<b>Split Completion Discard.</b> Set if PCI 6540 discards a Split Completion, because the requester on the primary bus would not accept it. Values: 0 = Split Completion not discarded 1 = Split Completion discarded	Yes	Yes/Clr	0
19	<b>Unexpected Split Completion.</b> Set if an Unexpected Split Completion with a Requester ID equal to PCI 6540 Primary Bus, Device, and Function Numbers is received on the bridge primary bus. Values: 0 = No Unexpected Split Completion was received 1 = Unexpected Split Completion was received	Yes	Yes/Clr	0
20	<b>Split Request Delay.</b> Set if PCI 6540 terminates a Split Completion on the primary bus, with Retry or Disconnect at the next ADB, because the bridge buffers are full. Used by algorithms that optimize the upstream Split Transaction Commitment Limit register setting. Values: 0 = Bridge accepted all Split Completions 1 = Bridge terminated a Split Completion with Retry or Disconnect	Yes	Yes/Clr	0
21	<b>Split Completion Overrun.</b> Set when PCI 6540 receives a request to forward a transaction on the primary bus, but cannot because there is insufficient space within the limit specified in the PCI-X Upstream Split Transaction register Split Transaction Commitment Limit bits (PCIXUPSTR[31:16]; PCI:F8h). Values: 0 = Bridge did not delay a Split Request 1 = Bridge delayed a Split Request	Yes	Yes/Clr	0
31:22	<b>Reserved.</b>	Yes	No	0h



**Register 6-108. (PCIXUPSTR; PCI:F8h) PCI-X Upstream Split Transaction**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Split Transaction Capacity.</b> PCI 6540 stores Split Completions for Memory Reads in the same buffer as Split Completions for I/O and Configuration Reads and Writes. Indicates the buffer size, in ADQ numbers, for storing Split Completions for Memory Reads for requesters on the secondary bus addressing completers on the primary bus.	Yes	No	32d
31:16	<b>Split Transaction Commitment Limit or Outstanding ADQ Limit.</b> Indicates the cumulative Sequence size for PCI-X Memory Read transactions forwarded by PCI 6540 from requesters on the secondary bus addressing completers on the primary bus. Also indicates upstream Split Transaction size of those types the PCI 6540 is allowed to commit to at one time.	Yes	Yes	32d

**Note:** *PCIXUPSTR controls the bridge buffer behavior for forwarding Split Transactions from a secondary requester to a primary bus completer.*

**Register 6-109. (PCIXDNSTR; PCI:FCh) PCI-X Downstream Split Transaction**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Split Transaction Capacity.</b> PCI 6540 stores Split Completions for Memory Reads in the same buffer as Split Completions for I/O and Configuration Reads and Writes. Indicates the buffer size, in ADQ numbers, for storing Split Completions for Memory Reads for requesters on the primary bus addressing completers on the secondary bus.	Yes	No	32d
31:16	<b>Split Transaction Commitment Limit or Outstanding ADQ Limit.</b> Indicates the cumulative Sequence size for PCI-X Memory Read transactions forwarded by PCI 6540 from requesters on the secondary bus addressing completers on the primary bus. Also indicates downstream Split Transaction size of those types the PCI 6540 is allowed to commit to at one time.	Yes	Yes	32d

**Note:** *PCIXDNSTR controls the bridge buffer behavior for forwarding Split Transactions from a primary bus requester to a secondary bus completer.*

## **6.2 PCI CONFIGURATION REGISTER ADDRESS MAPPING—NON-TRANSPARENT MODE**

### **6.2.1 PCI Configuration Register Address Mapping 00h to 7Fh—Non-Transparent Mode**

The PCI 6540 can be configured to operate as a transparent or non-transparent PCI-to-PCI bridge. Modes are selectable through the TRANS# pin. When TRANS# is high, the PCI 6540 is configured to operate in Non-Transparent mode.

In Non-Transparent mode, the PCI 6540 uses Header Type 0 for the PCI Configuration registers. Table 6-3 lists the register mapping and offsets used to access those registers from primary or secondary ports.

Registers listed with a PCI offset or address are accessed by standard PCI Type 0 Configuration accesses. The primary port retains full access to its own registers at offsets 00h to 40h and Read-Only access to the secondary port Configuration registers at offsets 40h to 7Fh. Similarly, the secondary port retains full access to its own registers at offsets 00h to 40h and Read-Only access to the primary port Configuration registers at offsets 40h to 7Fh.

Table 6-5. PCI Configuration Register Address Mapping 00h – 7Ch—Non-Transparent Mode

PCI Configuration Register Address		To ensure software compatibility with other versions of the PCI 6540 family and to ensure compatibility with future enhancements, write 0 to all unused bits.								PCI Writable <sup>3</sup>	Serial EEPROM Writable <sup>3</sup>
Primary Offset	Secondary Offset	31	24	23	16	15	8	7	0		
00h	40h	Device ID <sup>1, 2</sup>				Vendor ID <sup>1, 2</sup>				Yes	Yes
04h	44h	Primary Status				Primary Command				Yes	No
08h	48h	Class Code <sup>1, 2</sup>						Revision ID <sup>2</sup>		Yes	Yes
0Ch	4Ch	Built-In Self-Test ( <i>Not Supported</i> )	Header Type <sup>1, 2</sup>		Primary Latency Timer		Primary Cache Line Size		Yes	Yes	
10h	50h	Downstream I/O or Memory BAR 0								Yes	No
14h	54h	Downstream Memory BAR 1								Yes	No
18h	58h	Downstream Memory BAR 2 or Downstream Memory BAR 1 Upper 32 Bits								Yes	No
1Ch – 2Bh	5Ch – 6Bh	<i>Reserved</i>								No	No
2Ch	6Ch	Subsystem ID <sup>1, 2</sup>				Subsystem Vendor ID <sup>1, 2</sup>				Yes	Yes
30h	70h	<i>Reserved</i>								No	No
34h	74h	<i>Reserved</i>						New Capability Pointer <sup>2</sup>		No	No
38h	78h	<i>Reserved</i>								No	No
3Ch	7Ch	Primary Maximum Latency <sup>1</sup>	Primary Minimum Grant <sup>1</sup>		Primary Interrupt Pin		Primary Interrupt Line		Yes	No	
40h	00h	Device ID <sup>1, 2</sup>				Vendor ID <sup>1, 2</sup>				Yes	Yes
44h	04h	Secondary Status				Secondary Command				Yes	No
48h	08h	Class Code <sup>1, 2</sup>						Revision ID <sup>2</sup>		Yes	Yes
4Ch	0Ch	Built-In Self-Test ( <i>Not Supported</i> )	Header Type <sup>1, 2</sup>		Secondary Latency Timer		Secondary Cache Line Size		Yes	Yes	
50h	10h	Upstream I/O or Memory BAR 0								Yes	No
54h	14h	Upstream Memory BAR 1								Yes	No
58h	18h	Upstream Memory BAR 2 or Upstream Memory BAR 1 Upper 32 Bits								Yes	No
5Ch – 6Bh	1Ch – 2Bh	<i>Reserved</i>								No	No
6Ch	2Ch	Subsystem ID <sup>1, 2</sup>				Subsystem Vendor ID <sup>1, 2</sup>				Yes	Yes
70h	30h	<i>Reserved</i>								No	No
74h	34h	<i>Reserved</i>						New Capability Pointer <sup>2</sup>		No	No
78h	38h	<i>Reserved</i>								No	No
7Ch	3Ch	Secondary Maximum Latency <sup>1</sup>	Secondary Minimum Grant <sup>1</sup>		Secondary Interrupt Pin		Secondary Interrupt Line		Yes	No	

1. Writable only when the Read-Only Registers Write Enable bit is set (HSSRRC[7]=1; PCI:9Ch). Refer to the individual register descriptions to determine which bits are writable.

2. Shared registers for primary and secondary ports.

3. Refer to the individual register descriptions to determine which bits are writable.

## 6.2.2 Primary Configuration Registers—Non-Transparent Mode

Except for the declared shared registers in Table 6-3, primary and secondary ports have independent registers at offsets 00h to 3Fh.

Non-Transparent Configuration spaces at offsets 80h to FFh (Table 6-5) are accessible by primary and secondary masters. To avoid corruption by other masters, the PCI 6540 implements a Non-Transparent Configuration Ownership Semaphore register (NTCOS; PCI:D2h). Use this register for Configuration Writes to the registers listed in Table 6-5.

**Note:** To prevent unintended corruption of Configuration registers, use the correct byte size when accessing the Configuration registers.

### 6.2.2.1 Primary Port PCI Type 0 Header Registers

Register 6-110. (PCIIDR; Primary PCI:00h, Secondary PCI:40h) PCI Configuration ID

Bit	Description	Read	Write	Value after Reset
15:0	<b>Vendor ID.</b> Identifies device manufacturer. Defaults to PCI-SIG-issued PLX Vendor ID, 3388h, if a blank or no serial EEPROM is present.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	3388h
31:16	<b>Device ID.</b> Identifies particular device. Defaults to PLX PCI 6540 part number (0029h), if a blank or no serial EEPROM is present. <b>Note:</b> In Transparent mode, defaults to 0028h.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0029h

**Note:** PCIIDR is shared by the primary and secondary ports.

Register 6-111. (PCICR; Primary PCI:04h, Secondary PCI:44h) Primary PCI Command

Bit	Description	Read	Write	Value after Reset
0	<b>I/O Space Enable.</b> Controls bridge response to I/O accesses on primary interface. Values: 0 = Ignores I/O transactions 1 = Enables response to I/O transactions	Yes	Primary: Yes Secondary: No	0
1	<b>Memory Space Enable.</b> Controls bridge response to Memory accesses on primary interface. Values: 0 = Ignores Memory transactions 1 = Enables response to Memory transactions	Yes	Primary: Yes Secondary: No	0
2	<b>Master Enable.</b> Controls bridge ability to operate as a master on primary interface. Values: 0 = Does not initiate transactions on primary interface, and disables response to Memory or I/O transactions on secondary interface 1 = Enables the bridge to operate as a master on primary interface	Yes	Primary: Yes Secondary: No	0
3	<b>Special Cycle Enable. <i>Not Supported.</i></b>	Yes	No	0
4	<b>Memory Write and Invalidate Enable. <i>Not Supported.</i></b>	Yes	No	0
5	<b>VGA Palette Snoop Enable.</b> Controls bridge response to VGA-compatible Palette accesses. Values: 0 = Ignores VGA Palette accesses on primary interface 1 = Enables response to VGA Palette writes on primary interface (I/O address AD[9:0]=3C6h, 3C8h, and 3C9h)	Yes	Primary: Yes Secondary: No	0
6	<b>Parity Error Response Enable.</b> Controls bridge response to Parity errors. Values: 0 = Ignores Parity errors 1 = Performs normal parity checking	Yes	Primary: Yes Secondary: No	0
7	<b>Wait Cycle Control.</b> If set to 1, PCI 6540 performs address/data stepping.	Yes	Primary: Yes Secondary: No	1
8	<b>P_SERR# Enable.</b> Controls enable for the Primary System Error (P_SERR#) pin. Values: 0 = Disables P_SERR# driver 1 = Enables P_SERR# driver	Yes	Primary: Yes Secondary: No	0
9	<b>Fast Back-to-Back Enable.</b> Controls bridge ability to generate Fast Back-to-Back transactions to various devices on primary interface. Values: 0 = No Fast Back-to-Back transactions 1 = <b>Reserved</b> ; PCI 6540 does not generate Fast Back-to-Back cycles	Yes	Primary: Yes Secondary: No	0
15:10	<b>Reserved.</b>	Yes	No	0h

Register 6-112. (PCISR; Primary PCI:06h, Secondary PCI:46h) Primary PCI Status

Bit	Description	Read	Write	Value after Reset
3:0	<b>Reserved.</b>	Yes	No	0h
4	<b>New Capability Functions Support.</b> Writing 1 supports New Capabilities Functions. The New Capability Function ID is located at the PCI Configuration space offset determined by the New Capabilities linked list pointer value at CAP_PTR; Primary PCI:34h, Secondary PCI:74h.	Yes	No	1
5	<b>66 MHz-Capable.</b> If set to 1, this device supports a 66 MHz PCI clock environment.	Yes	No	1
6	<b>UDF.</b> No User-Definable Features.	Yes	No	0
7	<b>Fast Back-to-Back Capable.</b> Fast Back-to-Back write capable on primary port.	Yes	No	0
8	<b>Data Parity Error Detected.</b> Set when the following conditions are met: <ul style="list-style-type: none"> <li>• P_PERR# is asserted, and</li> <li>• Command register Parity Error Response Enable bit is set (PCICR[6]=1; Primary PCI:04h, Secondary PCI:44h)</li> </ul> Writing 1 clears bit to 0.	Yes	Primary: Yes/Clr Secondary: No	0
10:9	<b>DEVSEL# Timing.</b> Reads as 01b to indicate PCI 6540 responds no slower than with medium timing.	Yes	No	01b
11	<b>Signaled Target Abort.</b> Set by a target device whenever a Target Abort cycle occurs. Writing 1 clears bit to 0.	Yes	Primary: Yes/Clr Secondary: No	0
12	<b>Received Target Abort.</b> Set to 1 by the PCI 6540 when transactions are terminated with Target Abort. Writing 1 clears bit to 0.	Yes	Primary: Yes/Clr Secondary: No	0
13	<b>Received Master Abort.</b> Set to 1 by the PCI 6540 when transactions are terminated with Master Abort. Writing 1 clears bit to 0.	Yes	Primary: Yes/Clr Secondary: No	0
14	<b>Signaled System Error.</b> Set when P_SERR# is asserted. Writing 1 clears bit to 0.	Yes	Primary: Yes/Clr Secondary: No	0
15	<b>Detected Parity Error.</b> Set when a Parity error is detected, regardless of the Parity Error Response Enable bit state (PCICR[6]=X; Primary PCI:04h, Secondary PCI:44h). Writing 1 clears bit to 0.	Yes	Primary: Yes/Clr Secondary: No	0

**Register 6-113. (PCIREV; Primary PCI:08h, Secondary PCI:48h) PCI Revision ID**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Revision ID.</b> PCI 6540 revision.	Yes	No	1h

**Note:** PCIREV is shared by the primary and secondary ports.

**Register 6-114. (PCICCR; Primary PCI:09h – 0Bh, Secondary PCI:49h – 4Bh) PCI Class Code**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Register Level Programming Interface.</b> None defined.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0h
15:8	<b>Subclass Code.</b> PCI-to-PCI bridge (Transparent mode) or other bridge device (Non-Transparent mode).	Yes	Only if HSSRRC[7]=1; Serial EEPROM	Transparent mode = 04h. Non-Transparent mode = 80h
23:16	<b>Base Class Code.</b> Bridge device.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	06h

**Note:** PCICCR is shared by the primary and secondary ports.

**Register 6-115. (PCICLSR; Primary PCI:0Ch, Secondary PCI:4Ch) Primary PCI Cache Line Size**

Bit	Description	Read	Write	Value after Reset
7:0	<p><b>System Cache Line Size.</b> Specified in units of 32-bit words (Dwords). Only cache line sizes which are a power of two are valid. Maximum value is 20h. For values greater than 20h, PCI 6540 operates as if PCICLSR is programmed with value of 08h.</p> <p>Used when terminating Memory Write and Invalidate transactions. Memory Read prefetching is controlled by the Prefetch Count registers.</p>	Yes	Primary: Yes Secondary: No	0h

**Register 6-116. (PCILTR; Primary PCI:0Dh, Secondary PCI:4Dh) Primary PCI Bus Latency Timer**

Bit	Description	Read	Write	Value after Reset
7:0	<p><b>Primary PCI Bus Latency Timer.</b> Specifies the amount of time (in units of PCI Bus clocks) the PCI 6540, as a bus master, can burst data on the primary PCI Bus.</p>	Yes	Primary: Yes Secondary: No	0h

**Register 6-117. (PCIHTR; Primary PCI:0Eh, Secondary PCI:4Eh) PCI Header Type**

Bit	Description	Read	Write	Value after Reset
6:0	<p><b>Configuration Layout Type.</b> Specifies the layout of registers 10h to 3Fh in Configuration space. Header Type 0 is defined for PCI devices other than PCI-to-PCI bridges (Header Type 1) and Cardbus bridges (Header Type 2).</p> <p><b>Note:</b> Default value is 1h in Transparent mode.</p>	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0h
7	<p><b>Multi-Function Device.</b> Value of 1 indicates multiple (up to eight) functions (logical devices), each containing its own, individually addressable Configuration space, 64 Dwords in size.</p>	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0

**Note:** PCIHTR is shared by the primary and secondary ports.

**Register 6-118. (PCIBISTR; Primary PCI:0Fh, Secondary PCI:4Fh) PCI Built-In Self-Test**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Built-In Self Test (BIST).</b> <i>Not Supported.</i>	Yes	No	0h



Register 6-119. (PCIBAR0; Primary PCI:10h, Secondary PCI:50h) PCI Downstream I/O or Memory BAR 0

Bit	Description	Read	Write	Value after Reset
0	<b>Memory Space Indicator.</b> Writing 0 indicates the register maps into Memory space. Writing 1 indicates the register maps into I/O space.	Yes	Primary: By way of DWNBAR0MSK[6]* Secondary: No	0
2:1	<b>Register Location (If Memory Space).</b> When mapped into I/O space (PCIBAR0[0]=1), bits [3:1] are <b>reserved</b> . Therefore, I/O Base addresses must always be mapped to a 16-byte boundary. Values: 00b = Locate anywhere in 32-bit Memory Address space 01b = <b>Reserved</b>	Yes	No	00b
3	<b>Prefetchable (If Memory Space).</b> Writing 1 indicates there are no side effects on reads. When mapped into I/O space (PCIBAR0[0]=1), bits [3:1] are <b>reserved</b> . Therefore, I/O Base addresses must always be mapped to a 16-byte boundary. Reflects XB_MEM pin status.	Yes	Primary: By way of DWNBAR0MSK[7]* Secondary: No	XB_MEM
31:4	<b>Base Address.</b> Base address for downstream accesses.	Yes	Primary: Yes Secondary: No	0h

**Note:** \* On the primary bus, the value can be changed or written through the Extended Register Index and Data registers (EXTRIDX; PCI:D3h and EXTRDATA; PCI:D4h, respectively), as detailed in Section 6.2.4.14.

Register 6-120. (PCIBAR1; Primary PCI:14h, Secondary PCI:54h) PCI Downstream Memory BAR 1

Bit	Description	Read	Write	Value after Reset
0	<b>Memory Space Indicator.</b> Writing 0 indicates the register maps into Memory space.	Yes	No	0
2:1	<b>Register Location.</b> Values: 00b = Locate anywhere in 32-bit Memory Address space 01b = <b>Reserved</b> 10b = Locate anywhere in 64-bit PCI Address space Only PCIBAR1[2] can be written by way of DWNBAR1MSK[14]. PCIBAR1[1] is Read-Only.	Yes	Primary: By way of DWNBAR1MSK[14]* Secondary: No	00b
3	<b>Prefetchable.</b> Writing 1 indicates there are no side effects on reads. Reflects XB_MEM pin status.	Yes	Primary: By way of DWNBAR1MSK[15]* Secondary: No	XB_MEM
31:4	<b>Base Address.</b> Base address for downstream accesses.	Yes	Primary: Yes Secondary: No	0h

**Note:** \* On the primary bus, the value can be changed or written through the Extended Register Index and Data registers (EXTRIDX; PCI:D3h and EXTRDATA; PCI:D4h, respectively), as detailed in Section 6.2.4.14.

**Register 6-121. (PCIBAR2; Primary PCI:18h, Secondary PCI:58h) PCI Downstream Memory BAR 2 or Downstream Memory BAR 1 Upper 32 Bits**

Bit	Description	Read	Write	Value after Reset
0	<b>Memory Space Indicator.</b> Writing 0 indicates the register maps into Memory space.	Yes	Primary: No: DWNBAR1MSK[14]=0 Yes*: DWNBAR1MSK[14]=1 Secondary: No	0
2:1	<b>Register Location.</b> Values: 00b = Locate anywhere in 32-bit Memory Address space 01b = <b>Reserved</b>	Yes	Primary: No: DWNBAR1MSK[14]=0 Yes*: DWNBAR1MSK[14]=1 Secondary: No	00b
3	<b>Prefetchable.</b> Writing 1 indicates there are no side effects on reads. DWNBAR2MSK[23] can only be used to set the space as prefetchable if DWNBAR1MSK[14]=0. Reflects XB_MEM pin status.	Yes	Primary: By way of DWNBAR2MSK[23]* Secondary: No	XB_MEM
31:4	<b>Base Address.</b> Downstream access Base address.	Yes	Yes	0h

**Notes:** \* On the primary bus, the DWNBARxMSK value can be changed or written through the Extended Register Index and Data registers (EXTRIDX; PCI:D3h and EXTRDATA; PCI:D4h, respectively), as detailed in Section 6.2.4.14.

When using 64-bit addressing (DWNBAR1MSK[14]=1) for PCIBAR1, PCIBAR2 becomes the upper 32 bits of the PCIBAR1 address and is accessed using normal Type 0 Configuration accesses.

**Register 6-122. (PCISVID; Primary PCI:2Ch, Secondary PCI:6Ch) PCI Subsystem Vendor ID**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Subsystem Vendor ID.</b> Unique add-in board Vendor ID. PCI-SIG-issued PLX Vendor ID (3388h), if a blank or no serial EEPROM is present.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	3388h

**Note:** PCISVID is shared by the primary and secondary ports.

**Register 6-123. (PCISID; Primary PCI:2Eh, Secondary PCI:6Eh) PCI Subsystem ID**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Subsystem ID.</b> Unique add-in board Device ID. Defaults to PLX PCI 6540 part number (0029h), if a blank or no serial EEPROM is present. <b>Note:</b> In Transparent mode, defaults to 0028h.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0029h

**Note:** PCISID is shared by the primary and secondary ports.

**Register 6-124. (CAP\_PTR; Primary PCI:34h, Secondary PCI:74h) New Capability Pointer**

Bit	Description	Read	Write	Value after Reset
7:0	<b>New Capability Pointer.</b> Provides an offset into PCI Configuration space for the Power Management capability location in the New Capabilities Linked List.	Yes	No	DCh
31:8	<b>Reserved.</b>	Yes	No	0h

**Note:** CAP\_PTR is shared by the primary and secondary ports.

**Register 6-125. (PCIILR; PCI:3Ch, Secondary PCI:7Ch) Primary PCI Interrupt Line**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Interrupt Line Routing Value.</b> Indicates which input of the system interrupt controller(s) is connected to the PCI 6540 interrupt line.	Yes	Yes	0h

**Register 6-126. (PCIIPR; PCI:3Dh, Secondary PCI:7Dh) Primary PCI Interrupt Pin**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Interrupt Pin.</b> Indicates which interrupt pin the PCI 6540 uses. Value: 1h = Interrupt pin P_INTA#	Yes	No	1h

**Register 6-127. (PCIPMGR; PCI:3Eh, Secondary PCI:7Eh) Primary PCI Minimum Grant**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Primary Min_Gnt.</b> Specifies how long a burst period the PCI 6540 needs.	Yes	Only if HSSRRC[7]=1	0h

**Register 6-128. (PCIPMLR; PCI:3Fh, Secondary PCI:7Fh) Primary PCI Maximum Latency**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Primary Max_Lat.</b> Specifies how often PCI 6540 must gain access to the primary PCI Bus.	Yes	Only if HSSRRC[7]=1	0h

## 6.2.2.2 Secondary Port PCI Type 0 Header Registers

**Register 6-129. (PCIIDR; Primary PCI:40h, Secondary PCI:00h) PCI Configuration ID**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Vendor ID.</b> Identifies PCI 6540 manufacturer. Defaults to PCI-SIG-issued PLX Vendor ID (3388h), if a blank or no serial EEPROM is present.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	3388h
31:16	<b>Device ID.</b> Identifies the particular device. Defaults to PLX PCI 6540 part number (0029h), if a blank or no serial EEPROM is present. <b>Note:</b> <i>In Transparent mode, defaults to 0028h.</i>	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0029h

**Note:** *PCIIDR is shared by the primary and secondary ports.*

**Register 6-130. (PCISCR; Primary PCI:44h, Secondary PCI:04h) Secondary PCI Command**

Bit	Description	Read	Write	Value after Reset
0	<b>I/O Space Enable.</b> Controls bridge response to I/O accesses on secondary interface. Values: 0 = Ignores I/O transactions 1 = Enables response to I/O transactions	Yes	Primary: No Secondary: Yes	0
1	<b>Memory Space Enable.</b> Controls bridge response to Memory accesses on secondary interface. Values: 0 = Ignores Memory transactions 1 = Enables response to Memory transactions	Yes	Primary: No Secondary: Yes	0
2	<b>Master Enable.</b> Controls bridge ability to operate as a master on secondary interface. Values: 0 = Does not initiate transactions on secondary interface and disables response to Memory or I/O transactions on primary interface 1 = Enables the bridge to operate as a master on secondary interface	Yes	Primary: No Secondary: Yes	0
3	<b>Special Cycle Enable. <i>Not Supported.</i></b>	Yes	No	0
4	<b>Memory Write and Invalidate Enable. <i>Not Supported.</i></b>	Yes	No	0
5	<b>VGA Palette Snoop Enable.</b> Controls bridge response to VGA-compatible Palette accesses. Values: 0 = Ignores VGA Palette accesses on secondary interface 1 = Enables response to VGA Palette writes on secondary interface (I/O address AD[9:0]=3C6h, 3C8h, and 3C9h)	Yes	Primary: No Secondary: Yes	0
6	<b>Parity Error Response Enable.</b> Controls bridge response to Parity errors. Values: 0 = Ignores Parity errors 1 = Performs normal parity checking	Yes	Primary: No Secondary: Yes	0
7	<b>Wait Cycle Control.</b> If set to 1, PCI 6540 performs address/data stepping.	Yes	Primary: No Secondary: Yes	1

Register 6-130. (PCISCR; Primary PCI:44h, Secondary PCI:04h) Secondary PCI Command (Continued)

Bit	Description	Read	Write	Value after Reset
8	<b>S_SERR# Enable.</b> Controls enable for secondary System Error (S_SERR#) pin. Values: 0 = Disables S_SERR# driver 1 = Enables S_SERR# driver	Yes	Primary: No Secondary: Yes	0
9	<b>Fast Back-to-Back Enable.</b> Controls bridge ability to generate Fast Back-to-Back transactions to various devices on secondary interface. Values: 0 = No Fast Back-to-Back transactions 1 = <b>Reserved</b> ; PCI 6540 does not generate Fast Back-to-Back cycles	Yes	Primary: No Secondary: Yes	0
15:10	<b>Reserved.</b>	Yes	No	0h

Register 6-131. (PCISSR; Primary PCI:46h, Secondary PCI:06h) Secondary PCI Status

Bit	Description	Read	Write	Value after Reset
3:0	<b>Reserved.</b>	Yes	No	0h
4	<b>New Capability Functions Support.</b> Writing 1 supports New Capabilities Functions. The New Capability Function ID is located at the PCI Configuration space offset determined by the New Capabilities linked list pointer value at CAP_PTR; Primary PCI:34h, Secondary PCI:74h.	Yes	No	1
5	<b>66 MHz-Capable.</b> If set to 1, supports a 66 MHz PCI clock environment.	Yes	No	1
6	<b>UDF.</b> No user-definable features.	Yes	No	0
7	<b>Fast Back-to-Back Capable.</b> Fast Back-to-Back write capable on secondary port.	Yes	No	0
8	<b>Data Parity Error Detected.</b> Set when the following conditions are met: <ul style="list-style-type: none"> <li>S_PERR# is asserted, and</li> <li>Command register Parity Error Response Enable bit is set (PCICR[6]=1; Primary PCI:04h, Secondary PCI:44h)</li> </ul> Writing 1 clears bit to 0.	Yes	Primary: No Secondary: Yes/Clr	0
10:9	<b>DEVSEL# Timing.</b> Reads as 01b to indicate PCI 6540 responds no slower than with medium timing.	Yes	No	01b
11	<b>Signaled Target Abort.</b> Set by a target device when a Target Abort cycle occurs. Writing 1 clears bit to 0.	Yes	Primary: No Secondary: Yes/Clr	0

**Register 6-131. (PCISSR; Primary PCI:46h, Secondary PCI:06h) Secondary PCI Status (Continued)**

Bit	Description	Read	Write	Value after Reset
12	<b>Received Target Abort.</b> Set to 1 by PCI 6540 when transactions are terminated with Target Abort. Writing 1 clears bit to 0.	Yes	Primary: No Secondary: Yes/Clr	0
13	<b>Received Master Abort.</b> Set to 1 by PCI 6540 when transactions are terminated with Master Abort. Writing 1 clears bit to 0.	Yes	Primary: No Secondary: Yes/Clr	0
14	<b>Signaled System Error.</b> Set when S_SERR# is asserted. Writing 1 clears bit to 0.	Yes	Primary: No Secondary: Yes/Clr	0
15	<b>Detected Parity Error.</b> Set when a Parity error is detected, regardless of the Parity Error Response Enable bit state (PCICR[6]=X; Primary PCI:04h, Secondary PCI:44h). Writing 1 clears bit to 0.	Yes	Primary: No Secondary: Yes/Clr	0

**Register 6-132. (PCIREV; Primary PCI:48h, Secondary PCI:08h) PCI Revision ID**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Revision ID.</b> PCI 6540 revision.	Yes	No	1h

**Note:** PCIREV is shared by the primary and secondary ports.

**Register 6-133. (PCICCR; Primary PCI:49h – 4Bh, Secondary PCI:09h – 0Bh) PCI Class Code**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Register Level Programming Interface.</b> None defined.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0h
15:8	<b>Subclass Code.</b> PCI-to-PCI bridge (Transparent mode) or other bridge device (Non-Transparent mode).	Yes	Only if HSSRRC[7]=1; Serial EEPROM	Transparent mode = 04h. Non-Transparent mode = 80h
23:16	<b>Base Class Code.</b> Bridge device.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	06h

**Note:** PCICCR is shared by the primary and secondary ports.



**Register 6-134. (PCISCLSR; Primary PCI:4Ch, Secondary PCI:0Ch) Secondary PCI Cache Line Size**

Bit	Description	Read	Write	Value after Reset
7:0	<b>System Cache Line Size.</b> Specified in units of 32-bit words (Dwords). Only cache line sizes which are a power of two are valid. Maximum value is 20h. For values greater than 20h, PCI 6540 operates as if PCISCLSR is programmed with value of 08h. Used when terminating Memory Write and Invalidate transactions. Memory Read prefetching is controlled by the Prefetch Count registers.	Yes	Primary: No Secondary: Yes	0h

**Register 6-135. (PCISLTR; Primary PCI:4Dh, Secondary PCI:0Dh) Secondary PCI Bus Latency Timer**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Secondary PCI Bus Latency Timer.</b> Specifies the amount of time (in units of PCI Bus clocks) the PCI 6540, as a bus master, can burst data on the secondary PCI Bus.	Yes	Primary: No Secondary: Yes	0h

**Register 6-136. (PCIHTR; Primary PCI:4Eh, Secondary PCI:0Eh) PCI Header Type**

Bit	Description	Read	Write	Value after Reset
6:0	<b>Configuration Layout Type.</b> Specifies the layout of registers 10h to 3Fh in Configuration space. Header Type 0 is defined for PCI devices other than PCI-to-PCI bridges (Header Type 1) and Cardbus bridges (Header Type 2). <b>Note:</b> Default value is 1h in Transparent mode.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0h
7	<b>Multi-Function Device.</b> Value of 1 indicates multiple (up to eight) functions (logical devices), each containing its own, individually addressable Configuration space, 64 Dwords in size.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0

**Note:** PCIHTR is shared by the primary and secondary ports.

**Register 6-137. (PCIBISTR; Primary PCI:4Fh, Secondary PCI:0Fh) PCI Built-In Self-Test**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Built-In Self Test (BIST).</b> <i>Not Supported.</i>	Yes	No	0h

Register 6-138. (PCIUBAR0; Primary PCI:50h, Secondary PCI:10h) PCI Upstream I/O or Memory BAR 0

Bit	Description	Read	Write	Value after Reset
0	<b>Memory Space Indicator.</b> Writing 0 indicates the register maps into Memory space. Writing 1 indicates the register maps into I/O space.	Yes	Primary: No Secondary: By way of UPSBAR0MSK[6]*	0
2:1	<b>Register Location (If Memory Space).</b> When mapped into I/O space (PCIUBAR0[0]=1), bits [3:1] are <b>reserved</b> . Therefore, I/O Base addresses must always be mapped to a 16-byte boundary. Values: 00b = Locate anywhere in 32-bit Memory Address space 01b = <b>Reserved</b>	Yes	No	00b
3	<b>Prefetchable (If Memory Space).</b> Writing 1 indicates there are no side effects on reads. When mapped into I/O space (PCIUBAR0[0]=1), bits [3:1] are <b>reserved</b> . Therefore, I/O Base addresses must always be mapped to a 16-byte boundary. Reflects XB_MEM pin status.	Yes	Primary: No Secondary: By way of UPSBAR0MSK[7]*	XB_MEM
31:4	<b>Base Address.</b> Base address for upstream accesses.	Yes	Primary: No Secondary: Yes	0h

**Note:** \* On the secondary bus, the value can be changed or written through the Extended Register Index and Data registers (EXTRIDX; PCI:D3h and EXTRDATA; PCI:D4h, respectively), as detailed in Section 6.2.4.14.

**Register 6-139. (PCIUBAR1; Primary PCI:54h, Secondary PCI:14h) PCI Upstream Memory BAR 1**

Bit	Description	Read	Write	Value after Reset
0	<b>Memory Space Indicator.</b> Writing 0 indicates the register maps into Memory space.	Yes	No	0
2:1	<b>Register Location.</b> Values: 00b = Locate anywhere in 32-bit Memory Address space 01b = <b>Reserved</b> 10b = Locate anywhere in 64-bit PCI Address space Only PCIUBAR1[2] can be written by way of UPSBAR1MSK[14]. PCIUBAR1[1] is Read-Only.	Yes	Primary: No Secondary: By way of UPSBAR1MSK[14]*	00b
3	<b>Prefetchable.</b> Writing 1 indicates there are no side effects on reads. Reflects XB_MEM pin status.	Yes	Primary: No Secondary: By way of UPSBAR1MSK[15]*	XB_MEM
31:4	<b>Base Address.</b> Base address for upstream accesses.	Yes	Primary: No Secondary: Yes	0h

**Note:** \* On the secondary bus, the value can be changed or written through the Extended Register Index and Data registers (EXTRIDX; PCI:D3h and EXTRDATA; PCI:D4h, respectively), as detailed in Section 6.2.4.14.

**Register 6-140. (PCIUBAR2; Primary PCI:58h, Secondary PCI:18h) PCI Upstream Memory BAR 2 or Upstream Memory Bar 1 Upper 32 Bits**

Bit	Description	Read	Write	Value after Reset
0	<b>Memory Space Indicator.</b> Writing 0 indicates the register maps into Memory space.	Yes	Primary: No Secondary: No: UPSBAR1MSK[14]=0 Yes*: UPSBAR1MSK[14]=1	0
2:1	<b>Register Location.</b> Values: 00b = Locate anywhere in 32-bit Memory Address space 01b = <b>Reserved</b>	Yes	Primary: No Secondary: No: UPSBAR1MSK[14]=0 Yes*: UPSBAR1MSK[14]=1	00b
3	<b>Prefetchable.</b> Writing 1 indicates there are no side effects on reads. UPSBAR2MSK[23] can only be used to set the space as prefetchable if UPSBAR1MSK[14]=0. Reflects XB_MEM pin status.	Yes	Primary: No Secondary: By way of UPSBAR2MSK[23]*	XB_MEM
31:4	<b>Base Address.</b> Base address for upstream accesses.	Yes	Primary: No Secondary: Yes	0h

**Notes:** \* On the secondary bus, the UPSBARxMSK value can be changed or written through the Extended Register Index and Data registers (EXTRIDX; PCI:D3h and EXTRDATA; PCI:D4h, respectively), as detailed in Section 6.2.4.14.

When using 64-bit addressing (UPSBAR1MSK[14]=1) for PCIUBAR1, PCIUBAR2 becomes the upper 32 bits of the PCIUBAR1 address and is accessed using normal Type 0 Configuration accesses.

**Register 6-141. (PCISVID; Primary PCI:6Ch, Secondary PCI:2Ch) PCI Subsystem Vendor ID**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Subsystem Vendor ID.</b> Unique add-in board Vendor ID (3388h), if a blank or no serial EEPROM is present.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	3388h

**Note:** PCISVID is shared by the primary and secondary ports.

**Register 6-142. (PCISID; Primary PCI:6Eh, Secondary PCI:2Eh) PCI Subsystem ID**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Subsystem ID.</b> Unique add-in board Device ID. Defaults to PLX PCI 6540 part number (0029h), if a blank or no serial EEPROM is present. <b>Note:</b> In Transparent mode, defaults to 0028h.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0029h

**Note:** PCISID is shared by the primary and secondary ports.

**Register 6-143. (CAP\_PTR; Primary PCI:74h, Secondary PCI:34h) New Capability Pointer**

Bit	Description	Read	Write	Value after Reset
7:0	<b>New Capability Pointer.</b> Provides an offset into PCI Configuration space for the Power Management capability location in the New Capabilities Linked List.	Yes	No	DCh
31:8	<b>Reserved.</b>	Yes	No	0h

**Note:** CAP\_PTR is shared by the primary and secondary ports.

**Register 6-144. (PCISILR; PCI:7Ch, Secondary PCI:3Ch) Secondary PCI Interrupt Line**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Interrupt Line Routing Value.</b> Indicates which input of the system interrupt controller(s) is connected to the PCI 6540 interrupt line.	Yes	Yes	0h

**Register 6-145. (PCISIPR; PCI:7Dh, Secondary PCI:3Dh) Secondary PCI Interrupt Pin**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Interrupt Pin.</b> Indicates which interrupt pin the PCI 6540 uses. Value: 1h = Interrupt pin S_INTA#	Yes	No	1h

**Register 6-146. (PCISMGR; PCI:7Eh, Secondary PCI:3Eh) Secondary PCI Minimum Grant**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Secondary Min_Gnt.</b> Specifies how long a burst period the PCI 6540 needs.	Yes	Only if HSSRRC[7]=1	0h

**Register 6-147. (PCISMLR; PCI:7Fh, Secondary PCI:3Fh) Secondary PCI Maximum Latency**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Secondary Max_Lat.</b> Specifies how often PCI 6540 must gain access to the secondary PCI Bus.	Yes	Only if HSSRRC[7]=1	0h

### 6.2.3 PCI Shadow Configuration Registers—Non-Transparent Mode

Normally, registers at offsets 40h to 7Fh reflect the image of the opposite port PCI Configuration registers. In addition, Shadow registers at offsets 42h to 7Fh can be accessed by setting the Chip Control register

Transparent Access bit to 1 (CCNTRL[6]=1; PCI:D8h). These are shared Miscellaneous Control registers that are also used in Transparent mode.

These registers are restored to their default values upon P\_RSTIN# assertion and are accessible only by the primary port.

**Table 6-6. PCI Configuration Shadow Register Map—PCI Offset Used Only when CCNTRL[6]=1, Non-Transparent Mode**

PCI Configuration Register Address		To ensure software compatibility with other versions of the PCI 6540 family and to ensure compatibility with future enhancements, write 0 to all unused bits.								PCI Writable	Serial EEPROM Writable
Primary Offset	Secondary Offset	31	24	23	16	15	8	7	0		
40h	40h	Bridge Control				Reserved				Yes	No
44h	44h	Miscellaneous Options				Timeout Control		Primary Flow-Through Control		Yes	Yes
48h	48h	Secondary Incremental Prefetch Count		Primary Incremental Prefetch Count		Secondary Initial Prefetch Count		Primary Initial Prefetch Count		Yes	Yes
4Ch	4Ch	Buffer Control		Secondary Flow-Through Control		Secondary Maximum Prefetch Count		Primary Maximum Prefetch Count		Yes	Yes
50h	50h	Reserved		Test		Internal Arbiter Control				Yes	No
54h	54h	Serial EEPROM Data				Serial EEPROM Address		Serial EEPROM Control		Yes	No
58h	58h	Reserved								No	No
5Ch	5Ch	Reserved								No	No
60h	60h	Timer Counter				Timer Control		Reserved		Yes	No
74h – 7Fh	74h – 7Fh	Reserved								No	No

### 6.2.3.1 Primary Flow-Through Control Registers

Register 6-148. (BCNTRL; PCI:42h) Bridge Control

Bit	Description	Read	Write	Value after Reset
0	<b>Parity Error Response Enable.</b> Controls bridge response to Parity errors on secondary interface. Values: 0 = Ignores Address and Data Parity errors on secondary interface 1 = Enables Parity error reporting and detection on secondary interface	Yes	Yes	0
1	<b>S_SERR# Enable.</b> Controls S_SERR# forwarding to the primary interface. Values: 0 = Disables S_SERR# forwarding to primary interface 1 = Enables S_SERR# forwarding to primary interface	Yes	Yes	0
2	<b>ISA Enable.</b> Controls bridge response to ISA I/O addresses, which is limited to the first 64 KB. Values: 0 = Forwards I/O addresses in the range defined by the I/O Base and I/O Limit registers (PCIIOBAR; PCI:1Ch and PCIIO LMT; PCI:1Dh, respectively). 1 = Blocks 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 64 KB of I/O space that address the last 768 bytes in each 1-KB block. Secondary I/O transactions are forwarded upstream if the address falls within the last 768 bytes in each 1-KB block. Command Configuration register Master Enable bit must also be set (PCICR[2]=1; Primary PCI:04h, Secondary PCI:44h) to enable ISA.	Yes	Yes	0
3	<b>VGA Enable.</b> Controls bridge response to VGA-compatible addresses. Values: 0 = Does not forward VGA-compatible Memory and I/O addresses from primary to secondary 1 = Forwards VGA-compatible Memory and I/O address from primary to secondary, regardless of other settings	Yes	Yes	0
4	<b>Reserved.</b>	Yes	No	0
5	<b>Master Abort Mode.</b> Controls bridge behavior in response to Master Aborts on secondary interface. Values: 0 = Does not report Master Aborts (returns FFFF_FFFFh on reads and discards data on writes). 1 = Reports Master Aborts by signaling Target Abort. If the Master Abort is the result of a primary-to-secondary Posted Write cycle, P_SERR# is asserted (PCICR[8]=1; Primary PCI:04h, Secondary PCI:44h). <b>Note:</b> During Lock cycles, PCI 6540 ignores this bit, and completes the cycle as a Target Abort.	Yes	Yes	0
6	<b>Secondary Reset.</b> Forces S_RSTOUT# assertion on secondary interface. Values: 0 = Does not force S_RSTOUT# assertion 1 = Forces S_RSTOUT# assertion	Yes	Yes	0
7	<b>Fast Back-to-Back Enable.</b> Controls bridge ability to generate Fast Back-to-Back transactions to various devices on secondary interface. Values: 0 = No Fast Back-to-Back transaction 1 = <b>Reserved</b> ; PCI 6540 does not generate Fast Back-to-Back cycles	Yes	Yes	0



Register 6-148. (BCNTRL; PCI:42h) Bridge Control (Continued)

Bit	Description	Read	Write	Value after Reset
8	<b>Primary Master Timeout.</b> Sets the maximum number of PCI clocks for an initiator on the primary bus to repeat the Delayed Transaction request. Values: 0 = Timeout after $2^{15}$ PCI clocks 1 = Timeout after $2^{10}$ PCI clocks	Yes	Yes	0
9	<b>Secondary Master Timeout.</b> Sets the maximum number of PCI clocks for an initiator on the secondary bus to repeat the Delayed Transaction request. Values: 0 = Timeout after $2^{15}$ PCI clocks 1 = Timeout after $2^{10}$ PCI clocks	Yes	Yes	0
10	<b>Master Timeout Status.</b> Set to 1 when primary or secondary Master Timeout occurs. Writing 1 clears bit to 0.	Yes	Yes/Clr	0
11	<b>Master Timeout P_SERR# Enable.</b> Enables P_SERR# assertion during Master Timeout. Values: 0 = P_SERR# not asserted on Master Timeout 1 = P_SERR# asserted on primary or secondary Master Timeout	Yes	Yes	0
12	<b>Master Timeout S_SERR# Enable.</b> Enables S_SERR# assertion during Master Timeout. Values: 0 = S_SERR# not asserted on Master Timeout 1 = S_SERR# asserted on either primary or secondary Master Timeout	Yes	Yes	0
13	<b>P_SERR# Enable.</b> Controls P_SERR# forwarding to the secondary interface. Values: 0 = Disables P_SERR# forwarding to secondary port 1 = Enable the P_SERR# forwarding to secondary port	Yes	Yes	0
15:14	<b>Reserved.</b>	Yes	No	00b

Register 6-149. (PFTCR; PCI:44h) Primary Flow-Through Control

Bit	Description	Read	Write	Value after Reset
2:0	<p><b>Primary Posted Write Completion Wait Count.</b> Maximum number of clocks the PCI 6540 waits for Posted Write data from the initiator if delivering Write data in Flow-Through mode and the Internal Post Write queues are almost empty. If the count is exceeded without additional data from the initiator, the cycle to the target is terminated and completed later. Values:</p> <p>000b = De-asserts S_IRDY# and waits seven clocks for data on the primary bus before terminating cycle (same as 111b)</p> <p>001b = De-asserts S_IRDY# and waits one clock for data on the primary bus before terminating cycle</p> <p>...</p> <p>111b = De-asserts S_IRDY# and waits seven clocks for data on the primary bus before terminating cycle</p>	Yes	Yes; Serial EEPROM	000b
3	<b>Reserved.</b>	Yes	No	0
6:4	<p><b>Primary Delayed Read Completion Wait Count.</b> Maximum number of clocks the PCI 6540 waits for Delayed Read data from the target if returning Read data in Flow-Through mode and the Internal Delayed Read queue is almost full. If the count is exceeded without additional space in the queue, the cycle to the target is terminated, and completed when the initiator Retries the remainder of the cycle. Values:</p> <p>000b = De-asserts S_IRDY# and waits seven clocks for further data to be transferred to the primary bus before terminating cycle (same as 111b)</p> <p>001b = De-asserts S_IRDY# and waits one clock for additional space to be transferred to the primary bus before terminating cycle</p> <p>...</p> <p>111b = De-asserts S_IRDY# and waits seven clocks for further data to be transferred to the primary bus before terminating cycle</p>	Yes	Yes; Serial EEPROM	000b
7	<b>Reserved.</b>	Yes	No	0

## 6.2.3.2 Timeout Control Register

Register 6-150. (TOCNTRL; PCI:45h) Timeout Control

Bit	Description	Read	Write	Value after Reset
2:0	<b>Maximum Retry Counter Control.</b> Controls the maximum number of times the PCI 6540 Retries 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 on SERR# events enabled. Maximum number of Retries to timeout: 000b = $2^{24}$ 001b = $2^{18}$ 010b = $2^{12}$ 011b = $2^6$ 111b = $2^0$	Yes	Yes; Serial EEPROM	000b
7:3	<b>Reserved.</b>	Yes	No	0

### 6.2.3.3 Miscellaneous Options Register

Register 6-151. (MSCOPT; PCI:46h) Miscellaneous Options

Bit	Description	Read	Write	Value after Reset
0	<b>Write Completion Wait for PERR#.</b> If set to 1, PCI 6540 waits for target PERR# status before completing a Delayed Write transaction to the initiator.	Yes	Yes; Serial EEPROM	0
1	<b>Read Completion Wait for PAR.</b> If set to 1, PCI 6540 waits for target PAR status before completing a Delayed Read transaction to the initiator.	Yes	Yes; Serial EEPROM	0
2	<b>DRT Out-of-Order Enable.</b> If set to 1, PCI 6540 may return Delayed Read transactions in a different order than requested. Otherwise, Delayed Read transactions are returned in the same order as requested.	Yes	Yes; Serial EEPROM	0
3	<b>Generate Parity Enable.</b> Values: 0 = Passes along the PAR/PAR64 cycle as stored in the internal buffers 1 = PCI 6540, as a master, generates PAR/PAR64 to cycles going across the bridge	Yes	Yes; Serial EEPROM	0
6:4	<b>Address Step Control.</b> During Type 0 Configuration cycles, PCI 6540 drives the address for the number of clocks specified in these bits before asserting FRAME#. Defaults to 001b in Conventional PCI mode. In PCI-X mode, these bits have no effect and address stepping is hardcoded to four clocks. Values: 000b = Concurrently asserts FRAME# and drives the address on the bus 001b = Asserts FRAME# one clock after driving the address on the bus ... 111b = Asserts FRAME# seven clocks after driving the address on the bus	Yes	Yes; Serial EEPROM	001b
7	<b>P_REQ64#.</b> Value reflected by the P_REQ64# signal during reset.	Yes	Yes; Serial EEPROM	0

Register 6-151. (MSCOPT; PCI:46h) Miscellaneous Options (Continued)

Bit	Description	Read	Write	Value after Reset
8	<b>S_REQ64#.</b> Value reflected by the S_REQ64# signal during reset.	Yes	Yes; Serial EEPROM	1
9	<b>Prefetch Early Termination.</b> Values: 0 = Terminates prefetching at the Initial Prefetch Count if Flow Through is not achieved, and another Prefetching Read cycle is accepted by the PCI 6540 1 = Completes prefetching as programmed by the Prefetch Count registers, regardless of other outstanding prefetchable reads in the Transaction queue	Yes	Yes; Serial EEPROM	0
10	<b>Read Minimum Enable.</b> If set to 1, PCI 6540 only initiates Read cycles if there is sufficient available space in the FIFO as required by the Prefetch Count registers.	Yes	Yes; Serial EEPROM	0
15, 11	<b>Force 64-Bit Control.</b> If set and the target supports 64-bit transfers, 32-bit prefetchable reads or 32-bit Posted Memory Write cycles on one side are converted to 64-bit cycles on completion at the target bus. If set to 0, cycles are not converted. Values: 00b = Disable (default) 01b = Convert to 64-bit command on both ports 10b = Convert to 64-bit command on secondary port 11b = Convert to 64-bit command on primary port	Yes	Yes; Serial EEPROM	00b
12	<b>Memory Write and Invalidate Control.</b> Values: 0 = Retries Memory Write and Invalidate commands, if there is insufficient space for one cache line of data in the internal queues. 1 = Passes Memory Write and Invalidate commands, if there are one or more cache lines of FIFO space available. If there is insufficient space, completes as a Memory Write cycle.	Yes	Yes; Serial EEPROM	0
13	<b>Primary Lock Enable.</b> If set to 1, PCI 6540 follows the LOCK protocol on primary interface; otherwise, LOCK is ignored.	Yes	Yes; Serial EEPROM	0
14	<b>Secondary Lock Enable.</b> If set to 1, PCI 6540 follows the LOCK protocol on secondary interface; otherwise, LOCK is ignored.	Yes	Yes; Serial EEPROM	0

### 6.2.3.4 Prefetch Control Registers

Shadow registers 48h to 4Dh are the Prefetch Control registers, and used to fine-tune the PCI 6540 Memory Read prefetch behavior. (Refer to Section 18, “PCI Flow-Through Optimization,” for detailed descriptions of these registers.) These registers apply only if there are one or more ports operating in Conventional PCI mode.

**Register 6-152. (PITLPCNT; PCI:48h) Primary Initial Prefetch Count**

Bit	Description	Read	Write	Value after Reset
0	<i>Reserved.</i>	Yes	No	0
2:1	<p><b>PCI-X Primary Initial Prefetch Count.</b> When the primary bus is in PCI-X mode, these bits control the initial Prefetch Count on the primary bus during reads to Prefetchable Memory space. Value defines the cache line multiples for the initial prefetch count. Prefetch as follows:</p> <p>00b = 1 cache line (size defined by PCICLSR; Primary PCI:0Ch, Secondary PCI:4Ch)</p> <p>01b = 2 cache lines</p> <p>10b = 4 cache lines</p> <p>11b = 8 cache lines</p> <p>The Primary PCI-X 16 Cache Line Prefetch bit (PINPCNT[1]; PCI:4Ah Shadow register) enables prefetch of 16 cache lines.</p>	Yes	Yes; Serial EEPROM	00b
5:3	<p><b>PCI Primary Initial Prefetch Count.</b> When the primary bus is in Conventional PCI mode, these bits control the initial Prefetch Count on the primary bus during reads to Prefetchable Memory space. Prefetch as follows:</p> <p>001b = 08h Dwords</p> <p>010b = 10h Dwords</p> <p>101b = 20h Dwords</p> <p>Other values are <i>Reserved.</i></p>	Yes	Yes; Serial EEPROM	001b
7:6	<i>Reserved.</i>	Yes	No	00b

Register 6-153. (SITLPCNT; PCI:49h) Secondary Initial Prefetch Count

Bit	Description	Read	Write	Value after Reset
0	<b>Reserved.</b>	Yes	No	0
2:1	<b>PCI-X Secondary Initial Prefetch Count.</b> When the secondary bus is in PCI-X mode, these bits control the initial Prefetch Count on the secondary bus during reads initiated from the primary port. Value defines the cache line multiples for the initial Prefetch Count. Prefetch as follows: 00b = 1 cache line (size defined by PCICLSR; Primary PCI:0Ch, Secondary PCI:4Ch) 01b = 2 cache lines 10b = 4 cache lines 11b = 8 cache lines The Secondary PCI-X 16 Cache Line Prefetch bit (SINPCNT[1]; PCI:4Bh Shadow register) enables prefetch of 16 cache lines.	Yes	Yes; Serial EEPROM	00b
5:3	<b>PCI Secondary Initial Prefetch Count.</b> When the secondary bus is in Conventional PCI mode, these bits control the initial Prefetch Count on the secondary bus during reads initiated from the primary port. Prefetch as follows: 001b = 08h Dwords 010b = 10h Dwords 101b = 20h Dwords Other values are <b>Reserved.</b>	Yes	Yes; Serial EEPROM	001b
7:6	<b>Reserved.</b>	Yes	No	00b

Register 6-154. (PINPCNT; PCI:4Ah) Primary Incremental Prefetch Count

Bit	Description	Read	Write	Value after Reset
0	<b>Reserved.</b>	Yes	No	0
1	<b>Primary PCI-X 16 Cache Line Prefetch.</b> Value: 1 = Enables Primary PCI-X port to prefetch 16 cache lines. When set, the PCI-X Primary Initial Prefetch Count bits are ignored (PITLPCNT[2:1]; PCI:48h Shadow register).	Yes	Yes; Serial EEPROM	0
5:2	<b>Primary Incremental Prefetch Count.</b> Applies only to primary port Conventional PCI mode operation. These bits control the incremental Read Prefetch Count. When an entry's remaining prefetch Dword count falls below this value, the bridge prefetches an additional primary incremental Prefetch Count Dwords. Value is specified as a multiple of 4 x Dwords. The register value must not exceed half the value programmed in the Primary Maximum Prefetch Count register (PMAXPcnt; PCI:4Ch); otherwise, no incremental prefetch is performed. Prefetch as follows: 0000b = No incremental prefetch 0001b = 04h Dwords 0010b = 08h Dwords 0011b = 0Ch Dwords ... 1111b = 3Ch Dwords	Yes	Yes; Serial EEPROM	0000b
7:6	<b>Reserved.</b>	Yes	No	00b

Register 6-155. (SINCPCNT; PCI:4Bh) Secondary Incremental Prefetch Count

Bit	Description	Read	Write	Value after Reset
0	<b>Reserved.</b>	Yes	No	0
1	<b>Secondary PCI-X 16 Cache Line Prefetch.</b> Value: 1 = Enable secondary PCI-X port to prefetch 16 cache lines. When set, the PCI-X Secondary Initial Prefetch Count bits are ignored (SITLPCNT[2:1]; PCI:49h Shadow register).	Yes	Yes; Serial EEPROM	0
5:2	<b>Secondary Incremental Prefetch Count.</b> Applies only to secondary port Conventional PCI mode operation. Controls the incremental read prefetch count. When an entry's remaining prefetch Dword count falls below this value, the bridge prefetches an additional secondary incremental prefetch count Dwords. Value is specified as a multiple of 4 x Dwords. The register value must not exceed half the value programmed in the Secondary Maximum Prefetch Count register (SMAXPcnt; PCI:4Dh); otherwise, no incremental prefetch is performed. Prefetch as follows: 0000b = No incremental prefetch 0001b = 04h Dwords 0010b = 08h Dwords 0011b = 0Ch Dwords ... 1111b = 3Ch Dwords	Yes	Yes; Serial EEPROM	0000b
7:6	<b>Reserved.</b>	Yes	No	00b



**Register 6-156. (PMAXPNT; PCI:4Ch) Primary Maximum Prefetch Count**

Bit	Description	Read	Write	Value after Reset
5:0	<b>Primary Maximum Prefetch Count.</b> Limits the cumulative maximum count of prefetchable Dwords that are allocated to one entry on the primary bus when Flow Through for that entry was not achieved. Value should be an even number. Bit 0 is Read-Only and always 0. Value is specified in Dwords, except if 0 value is programmed, which sets the Primary Maximum Prefetch Count to its maximum value of 256 bytes. This feature applies only to PCI-to-PCI bridging. A PCI Read cycle causes a PCI request for the Maximum Count data.	Yes	Yes [5:1]; Serial EEPROM	20h
7:6	<i>Reserved.</i>	Yes	No	00b

**Register 6-157. (SMAXPNT; PCI:4Dh) Secondary Maximum Prefetch Count**

Bit	Description	Read	Write	Value after Reset
5:0	<b>Secondary Maximum Prefetch Count.</b> Limits the cumulative maximum count of prefetchable Dwords that are allocated to one entry on the secondary bus when Flow Through for that entry was not achieved. Value should be an even number. Bit 0 is Read-Only and always 0. Value is specified in Dwords, except if 0 value is programmed, which sets the Secondary Maximum Prefetch Count to its maximum value of 256 bytes. This feature applies only to PCI-to-PCI bridging. A PCI Read cycle causes a PCI request for the Maximum Count data.	Yes	Yes [5:1]; Serial EEPROM	20h
7:6	<i>Reserved.</i>	Yes	No	00b

### 6.2.3.5 Secondary Flow-Through Control Register

Register 6-158. (SFTCR; PCI:4Eh) Secondary Flow-Through Control

Bit	Description	Read	Write	Value after Reset
2:0	<b>Secondary Posted Write Completion Wait Count.</b> Maximum number of clocks the PCI 6540 waits for Posted Write data from the initiator if delivering Write data in Flow-Through mode and the Internal Post Write queues are almost empty. If the count is exceeded without additional data from the initiator, the cycle to the target is terminated and completed later. Values: 000b = De-asserts P_IRDY# and waits seven clocks for data on the secondary bus, before terminating cycle (same as 111b) 001b = De-asserts P_IRDY# and waits one clock for data on the secondary bus before terminating cycle ... 111b = De-asserts P_IRDY# and waits seven clocks for data on the secondary bus before terminating cycle	Yes	Yes; Serial EEPROM	000b
3	<b>Reserved.</b>	Yes	No	0
6:4	<b>Secondary Delayed Read Completion Wait Count.</b> Maximum number of clocks the PCI 6540 waits for Delayed Read data from the target if returning Read data in Flow-Through mode and the Internal Delayed Read queue is almost full. If the count is exceeded without additional space in the queue, the cycle to target is terminated, and completed when initiator Retries the remainder of the cycle. Values: 000b = De-asserts P_IRDY# and waits seven clocks for additional space to be transferred to the secondary bus before terminating cycle (same as 111b) 001b = De-asserts P_IRDY# and waits one clock for further data to be transferred to the secondary bus before terminating cycle ... 111b = De-asserts P_IRDY# and waits seven clocks for further data to be transferred to the secondary bus before terminating cycle	Yes	Yes; Serial EEPROM	000b
7	<b>Reserved.</b>	Yes	No	0

### 6.2.3.6 Buffer and Internal Arbiter Control Registers

**Register 6-159. (BUFCR; PCI:4Fh) Buffer Control**

Bit	Description	Read	Write	Value after Reset
7:0	<i>Reserved.</i>	Yes	No	0h

Register 6-160. (IACNTRL; PCI:50h) Internal Arbiter Control

Bit	Description	Read	Write	Value after Reset
0	<b>Low-Priority Group Fixed Arbitration.</b> If set to 1, the low-priority group uses fixed priority arbitration; otherwise, rotating priority arbitration is used.	Yes	Yes	0
1	<b>Low-Priority Group Arbitration Order.</b> Valid only when the low-priority arbitration group is set to a fixed arbitration scheme. Values: 0 = Priority decreases with bus master number. ( <i>For example, assuming Master 2 is set as the highest priority master, Master 3 retains higher priority than Master 4.</i> ) 1 = Priority increases with bus master number. ( <i>For example, assuming Master 2 is set as the highest priority master, Master 4 retains higher priority than Master 3.</i> ) This order is relative to the master with the highest priority for this group, as specified in IACNTRL[7:4].	Yes	Yes	0
2	<b>High-Priority Group Fixed Arbitration.</b> If set to 1, the high-priority group uses the fixed priority arbitration; otherwise, rotating priority arbitration is used.	Yes	Yes	0
3	<b>High-Priority Group Arbitration Order.</b> Valid only when the high-priority arbitration group is set to a fixed arbitration scheme. Values: 0 = Priority decreases with bus master number. ( <i>For example, assuming Master 2 is set as the highest priority master, Master 3 retains higher priority than Master 4.</i> ) 1 = Priority increases with bus master number. ( <i>For example, assuming Master 2 is set as the highest priority master, Master 4 retains higher priority than Master 3.</i> ) This order is relative to the master with the highest priority for this group, as specified in IACNTRL[11:8].	Yes	Yes	0
7:4	<b>Highest Priority Master in Low-Priority Group.</b> Controls which master in the low-priority group retains the highest priority. Valid only if the group uses the fixed arbitration scheme. Values: 0000b = Master 0 retains highest priority 0001b = Master 1 retains highest priority ... 1000b = PCI 6540 retains highest priority 1001b – 1111b = <b>Reserved</b>	Yes	Yes	0000b
11:8	<b>Highest Priority Master in High-Priority Group.</b> Controls which master in the high-priority group remains the highest priority. Valid only if the group uses the fixed arbitration scheme. Values: 0000b = Master 0 retains highest priority 0001b = Master 1 retains highest priority ... 1000b = PCI 6540 retains highest priority 1001b – 1111b = <b>Reserved</b>	Yes	Yes	0000b
15:12	<b>Bus Grant Parking Control.</b> Controls bus grant behavior during idle. Value: 0h = Indicates the last master granted is parked All other values are <b>Reserved</b> .	Yes	No	0h

## 6.2.3.7 Test and Serial EEPROM Registers

Register 6-161. (TEST; PCI:52h) Test

Bit	Description	Read	Write	Value after Reset
0	<b>Serial EEPROM Autoload Control.</b> If set to 1, disables serial EEPROM autoload.	Yes	Yes	0
1	<b>Fast Serial EEPROM Autoload.</b> If set to 1, speeds up serial EEPROM autoload.	Yes	Yes	0
2	<b>Serial EEPROM Autoload Status.</b> Serial EEPROM autoload status is set to 1 during autoload.	Yes	No	Status of Serial EEPROM autoload
3	<b>S_PLL_TEST.</b> When P_CLKOE input pin is set to 1 and this bit is set to 1, S_CLKO4 (derived from S_CLKIN) is divided by 4.	Yes	Yes	0
4	<b>DEV64#.</b> Reflects DEV64# pin status.	Yes	No	DEV64#
5	<b>S_CFN#.</b> Reflects S_CFN# pin status.	Yes	No	S_CFN#
6	<b>TRANS#.</b> Reflects TRANS# pin status.	Yes	No	TRANS#
7	<b>U_MODE.</b> Reflects U_MODE pin status.	Yes	No	U_MODE

Register 6-162. (EEPCNTRL; PCI:54h) Serial EEPROM Control

Bit	Description	Read	Write	Value after Reset
0	<b>Start.</b> Starts serial EEPROM Read or Write cycle. Bit is cleared when serial EEPROM load completes.	Yes	Yes	0
1	<b>Serial EEPROM Command.</b> Controls commands sent to the serial EEPROM. Values: 0 = Read 1 = Write	Yes	Yes	0
2	<b>Serial EEPROM Error.</b> Set to 1 if serial EEPROM ACK was not received during serial EEPROM cycle.	Yes	No	—
3	<b>Serial EEPROM Autoload Successful.</b> Set to 1 if serial EEPROM autoload occurred successfully after reset, with appropriate Configuration registers loaded with the values programmed in the serial EEPROM. If 0, the serial EEPROM autoload was unsuccessful or disabled.	Yes	No	—
5:4	<b>Reserved.</b>	Yes	No	00b
7:6	<b>Serial EEPROM Clock Rate.</b> Controls the serial EEPROM clock frequency. The serial EEPROM clock is derived from the primary PCI clock. Values: 00b = PCLK / 1024 01b = PCLK / 512 10b = PCLK / 128 11b = PCLK / 16 (Test mode)	Yes	Yes	01b

Register 6-163. (EEPADDR; PCI:55h) Serial EEPROM Address

Bit	Description	Read	Write	Value after Reset
0	<i>Reserved.</i>	Yes	No	—
7:1	<b>Serial EEPROM Address.</b> Serial EEPROM cycle word address.	Yes	Yes	—

Register 6-164. (EEPDATA; PCI:56h) Serial EEPROM Data

Bit	Description	Read	Write	Value after Reset
15:0	<b>Serial EEPROM Data.</b> Contains data to be written to the serial EEPROM. During reads, contains data received from the serial EEPROM after a Read cycle completes.	Yes	Yes	—

## 6.2.3.8 Timer Registers

Register 6-165. (TMRCNTRL; PCI:61h) Timer Control

Bit	Description	Read	Write	Value after Reset
0	<b>Timer Enable.</b> Set to start measurement of the approximate bus frequency on the primary or secondary interface. By default, bit is 0, and must be set to 1 to start the measurement. When set to 0 and then to 1, PCI 6540 starts counting up until it reaches the set count period. During this counting period, PCI 6540 Timer Counter (TMRcnt; PCI:62h Shadow register) counts the number of Timer Counter clocks.	Yes	Yes	0
2:1	<b>Timer Counter Clock Source Select:</b> Values: 00b = Primary PCI clock (P_CLKIN) 01b = Secondary PCI clock (S_CLKIN) 10b, 11b = <i>Reserved</i>	Yes	Yes	00b
3	<b>Timer Stop.</b> Timer stopped status bit. When the measurement is finished, this bit is set to 0, and then to 1. When starting a new measurement, this bit automatically restores to 0. Values: 0 = Timer running 1 = Timer stopped	Yes	No	0
5:4	<b>Count Period.</b> Values: 00b = 16 Reference clock high states 01b = 32 Reference clock high states 10b = 64 Reference clock high states 11b = 128 Reference clock high states	Yes	Yes	00b
7:6	<i>Reserved.</i>	Yes	No	00b

Register 6-166. (TMRcnt; PCI:62h) Timer Counter

Bit	Description	Read	Write	Value after Reset
15:0	<b>Timer Counter.</b> Automatically stops upon the count period setting in the Timer Control register (TMRCNTRL[7:4]; PCI:61h Shadow register). This counter can be enabled by setting the Timer Enable bit (TMRCNTRL[0]; PCI:61h Shadow register) first to 0, and then to 1.	Yes	No	0h

## 6.2.4 PCI Configuration Register Address Mapping 80h to FFh, Shadow and Extended Registers—Non-Transparent Mode

Registers 80h to FFh, Shadow and Extended registers, are shared registers and can be accessed by primary and secondary ports. Take care when accessing these registers by both primary and secondary port masters. When possible, use the built-in semaphore mechanism.

Registers listed with a PCI offset or address are accessed by standard PCI Type 0 Configuration accesses.

### 6.2.4.1 Configuration Registers 80h to FFh

Registers 80h to FFh, and the extended registers, are set to their default values upon PCI 6540 power-up. Subsequent PCI-X resets from P\_RSTIN# and/or S\_RSTIN# do not affect their values.

Table 6-7. PCI Configuration Shadow Register Address Mapping 80h – FFh—Non-Transparent Mode

PCI Configuration Register Address		To ensure software compatibility with other versions of the PCI 6540 family and to ensure compatibility with future enhancements, write 0 to all unused bits.								PCI Writable	Serial EEPROM Writable
Primary Offset	Secondary Offset	31	24	23	16	15	8	7	0		
80h	80h	Cross-Bridge Downstream Configuration Address								Yes	No
84h	84h	Cross-Bridge Downstream Configuration Data								Yes	No
88h	88h	Cross-Bridge Upstream Configuration Address								Yes	No
8Ch	8Ch	Cross-Bridge Upstream Configuration Data								Yes	No
90h	90h	<i>Reserved</i>	Cross-Bridge Configuration Access Ownership Status		Cross-Bridge Upstream Configuration Ownership Semaphore		Cross-Bridge Downstream Configuration Ownership Semaphore		Yes	No	
94h	94h	S_SERR# Event Disable	P_SERR# Event Disable		Clock Control				Yes	No	
98h	98h	GPIO[3:0] Input Data	GPIO[3:0] Output Enable		GPIO[3:0] Output Data		P_SERR# and S_SERR# Status		Yes	No	
9Ch	9Ch	GPIO[7:4] Input Data	GPIO[7:4] Output Enable		GPIO[7:4] Output Data		Hot Swap Switch and ROR Control		Yes	No	
A0h	A0h	GPIO[15:8] Input Data	GPIO[15:8] Output Enable		GPIO[15:8] Output Data		Power-Up Status		Yes	No	
A4h	A4h	Upstream Message 3	Upstream Message 2		Upstream Message 1		Upstream Message 0		Yes	No	
A8h	A8h	Downstream Message 3	Downstream Message 2		Downstream Message 1		Downstream Message 0		Yes	No	
ACh	ACh	<i>Reserved</i>	Message Signaled Interrupts Control		Next Capability Pointer (0h)		Message Signaled Interrupts Capability ID (05h)*		Yes	Yes	



Table 6-7. PCI Configuration Shadow Register Address Mapping 80h – FFh—Non-Transparent Mode

PCI Configuration Register Address		To ensure software compatibility with other versions of the PCI 6540 family and to ensure compatibility with future enhancements, write 0 to all unused bits.								PCI Writable	Serial EEPROM Writable
Primary Offset	Secondary Offset	31	24	23	16	15	8	7	0		
B0h	B0h	Message Signaled Interrupts Address								Yes	No
B4h	B4h	Message Signaled Interrupts Upper Address								Yes	No
B8h	B8h	Reserved				Message Signaled Interrupts Data				Yes	No
BCh	BCh	Reserved								No	No
C0h	C0h	Downstream Doorbell Interrupt Request				Downstream Doorbell Interrupt Enable				Yes	No
C4h	C4h	Upstream Doorbell Interrupt Request				Upstream Doorbell Interrupt Enable				Yes	No
C8h	C8h	Upstream Interrupt Enable		Downstream Interrupt Status		Downstream Doorbell Interrupt Status				Yes	No
CCh	CCh	Downstream Interrupt Enable		Upstream Interrupt Status		Upstream Doorbell Interrupt Status				Yes	No
D0h	D0h	Extended Register Index		Non-Transparent Configuration Ownership Semaphore		Reserved				Yes	No
D4h	D4h	Extended Register Data								Yes	No
D8h	D8h	Arbiter Control				Diagnostic Control		Chip Control		Yes	No
DCh	DCh	Power Management Capabilities*				Power Management Next Capability Pointer (E4h)		Power Management Capability ID (01h)		Yes	Yes
E0h	E0h	Power Management Data*		PMCSR Bridge Supports Extensions		Power Management Control/Status*				Yes	Yes
E4h	E4h	Reserved		Hot Swap Control/Status (0h)		Hot Swap Next Capability Pointer (E8h)		Hot Swap Control (Capability ID) (06h)		Yes	No
E8h	E8h	VPD Address (0h)				VPD Next Capability Pointer (F0h)		VPD Capability ID (03h)		Yes	No
ECh	ECh	VPD Data (0h)								Yes	No
F0h	F0h	PCI-X Command				PCI-X Next Capability Pointer (0h)		PCI-X Capability ID (07h)		Yes	No
F4h	F4h	PCI-X Bridge Status								Yes	No
F8h	F8h	PCI-X Upstream Split Transaction								Yes	No
FCh	FCh	PCI-X Downstream Split Transaction								Yes	No

**Notes:** \* Writable only when the Read-Only Registers Write Enable bit is set (HSSRRC[7]=1; PCI:9Ch). Refer to the individual register descriptions to determine which bits are writable.

Refer to the individual register descriptions to determine which bits are writable.

### 6.2.4.2 Cross-Bridge Configuration Access Control Registers

Registers 80h to 87h and 90h cannot be written from the downstream side. Registers 88h to 8Fh and 91h cannot be written from the upstream side. Configuration addresses should always be set up before accessing the Configuration data.

**Register 6-167. (XBDWNCA; PCI:80h) Cross-Bridge Downstream Configuration Address**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Downstream Configuration Address.</b> Data used as the downstream Configuration address.	Yes	Yes	0h

**Register 6-168. (XBDWNCD; PCI:84h) Cross-Bridge Downstream Configuration Data**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Downstream Configuration Data.</b> Data presented is used as the downstream Configuration Read/Write data.	Yes	Yes	0h

**Register 6-169. (XBUPSCA; PCI:88h) Cross-Bridge Upstream Configuration Address**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Upstream Configuration Address.</b> Data used as the upstream Configuration address.	Yes	Yes	0h

**Register 6-170. (XBUPSCD; PCI:8Ch) Cross-Bridge Upstream Configuration Data**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Upstream Configuration Data.</b> Data presented is used as the upstream Configuration Read/Write data.	Yes	Yes	0h

**Register 6-171. (XBDWNCOS; PCI:90h) Cross-Bridge Downstream Configuration Ownership Semaphore**

Bit	Description	Read	Write	Value after Reset
0	<p><b>Downstream Configuration Ownership.</b> Only the upstream master can access this bit, using byte-wide accesses. When read as 0 by the upstream interface intending to access Downstream Configuration registers, indicates that Downstream Configuration Address and Data registers are not owned and can be accessed. The Read operation automatically sets this bit to 1, indicating the bus is owned. The owner issues a Configuration Write 1 to clear this bit after use.</p> <p>If the bit is not cleared when another master wants to access the register, that master also checks this bit. The bit is sampled as 1, indicating the register is in use, and the master waits until the register is available for access.</p> <p>Software can check this semaphore mechanism status by way of XBCOS[0]; PCI:92h, without taking ownership.</p>	Yes	Yes/Clr	0
7:1	<i>Reserved.</i>	Yes	No	0h

**Register 6-172. (XBUPSCOS; PCI:91h) Cross-Bridge Upstream Configuration Ownership Semaphore**

Bit	Description	Read	Write	Value after Reset
0	<p><b>Upstream Configuration Ownership.</b> Only the downstream master can access this bit, using byte-wide accesses. When read as 0 by the downstream interface intending to access Upstream Configuration registers, indicates the Upstream Configuration Address and Data registers are not owned and can be accessed. The Read operation automatically sets this bit to 1, indicating the bus is owned. The owner issues a Configuration Write 1 to clear this bit after use.</p> <p>If the bit is not cleared when another master wants to access the register, that master also checks this bit. The bit is sampled as 1, indicating the register is in use, and the master waits until the register is available for access.</p> <p>Software can check this semaphore mechanism status by way of XBCOS[1]; PCI:92h, without taking ownership.</p>	Yes	Yes/Clr	0
7:1	<i>Reserved.</i>	Yes	No	0h

**Register 6-173. (XBCOS; PCI:92h) Cross-Bridge Configuration Ownership Status**

Bit	Description	Read	Write	Value after Reset
0	<b>Downstream Configuration Ownership Status.</b> Allows software to check the Downstream Configuration Ownership bit (XBDWNCOS[0]; PCI:90h) without setting it.	Yes	No	0
1	<b>Upstream Configuration Ownership Status.</b> Allows software to check the Upstream Configuration Ownership bit (XBUPSCOS[0]; PCI:91h) without setting it.	Yes	No	0
7:2	<i>Reserved.</i>	Yes	No	0h

### 6.2.4.3 Clock Control Register

Register 6-174. (CLKCNTRL; PCI:94h) Clock Control

Bit	Description	Read	Write	Value after Reset
1:0	<b>Clock 0 Disable.</b> If either bit is 0, S_CLKO0 is enabled. When both bits are 1, S_CLKO0 is disabled. Defaults to 00b if MSK_IN=1.	Yes	Yes	00b
3:2	<b>Clock 1 Disable.</b> If either bit is 0, S_CLKO1 is enabled. When both bits are 1, S_CLKO1 is disabled.	Yes	Yes	00b
5:4	<b>Clock 2 Disable.</b> If either bit is 0, S_CLKO2 is enabled. When both bits are 1, S_CLKO2 is disabled.	Yes	Yes	00b
7:6	<b>Clock 3 Disable.</b> If either bit is 0, S_CLKO3 is enabled. When both bits are 1, S_CLKO3 is disabled.	Yes	Yes	00b
8	<b>Clock 4 Disable.</b> If 0, S_CLKO4 is enabled. When 1, S_CLKO4 is disabled.	Yes	Yes	0
15:9	<b>Reserved.</b>	Yes	No	0h

## 6.2.4.4 System Error Event Registers

Register 6-175. (PSERRED; PCI:96h) P\_SERR# Event Disable

Bit	Description	Read	Write	Value after Reset
0	<b>Address Parity Error.</b> P_SERR# is asserted because an Address Parity error occurred on either side of the bridge.	Yes	Yes/Clr	0
1	<b>Posted Write Parity Error.</b> Controls PCI 6540 ability 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 bit is 0 and the Command register P_SERR# Enable bit is set (PCICR[8]=1; Primary PCI:04h, Secondary PCI:44h).	Yes	Yes	0
2	<b>Posted Memory Write Non-Delivery.</b> Controls PCI 6540 ability to assert P_SERR# when it is unable to deliver Posted Write data after 2 <sup>24</sup> attempts [or programmed Maximum Retry count (TOCNTRL[2:0]; PCI:45h Shadow register)]. P_SERR# is asserted if this event occurs when bit is 0 and the Command register P_SERR# Enable bit is set (PCICR[8]=1; Primary PCI:04h, Secondary PCI:44h).	Yes	Yes	0
3	<b>Target Abort during Posted Write.</b> Controls PCI 6540 ability to assert P_SERR# when it receives a Target Abort while attempting to deliver Posted Write data. P_SERR# is asserted if this event occurs when bit is 0 and the Command register P_SERR# Enable bit is set (PCICR[8]=1; Primary PCI:04h, Secondary PCI:44h).	Yes	Yes	0
4	<b>Master Abort on Posted Write.</b> Controls PCI 6540 ability to assert P_SERR# when it receives a Master Abort while attempting to deliver Posted Write data. P_SERR# is asserted if this event occurs when bit is 0 and the Command register P_SERR# Enable bit is set (PCICR[8]=1; Primary PCI:04h, Secondary PCI:44h).	Yes	Yes	0
5	<b>Delayed Configuration or I/O Write Non-Delivery.</b> Controls PCI 6540 ability to assert P_SERR# when it is unable to deliver Delayed Write data after 2 <sup>24</sup> attempts [or programmed Maximum Retry count attempts (TOCNTRL[2:0]; PCI:45h Shadow register)]. P_SERR# is asserted if this event occurs when bit is 0 and the Command register P_SERR# Enable bit is set (PCICR[8]=1; Primary PCI:04h, Secondary PCI:44h).	Yes	Yes	0
6	<b>Delayed Read-No Data from Target.</b> Controls PCI 6540 ability to assert P_SERR# when it is unable to transfer Read data from the target after 2 <sup>24</sup> attempts [or programmed Maximum Retry count attempts (TOCNTRL[2:0]; PCI:45h Shadow register)]. P_SERR# is asserted if this event occurs when bit is 0 and Command register P_SERR# Enable bit is set (PCICR[8]=1; Primary PCI:04h, Secondary PCI:44h).	Yes	Yes	0
7	<b>Posted Write Data Parity Error.</b> P_SERR# is asserted because a Posted Write Data Parity error occurred on the target bus.	Yes	Yes/Clr	0

Register 6-176. (SSERRED; PCI:97h) S\_SERR# Event Disable

Bit	Description	Read	Write	Value after Reset
0	<b>Address Parity Error.</b> S_SERR# is asserted because an Address Parity error occurred on either side of the bridge.	Yes	Yes/Clr	0
1	<b>Posted Write Parity Error.</b> Controls PCI 6540 ability to assert S_SERR# when a Data Parity error is detected on the target bus during a Posted Write transaction. S_SERR# is asserted if this event occurs when bit is 0 and the Command register SERR# Enable bit is set (PCISCR[8]=1; Primary PCI:44h, Secondary PCI:04h).	Yes	Yes	0
2	<b>Posted Memory Write Non-Delivery.</b> Controls PCI 6540 ability to assert S_SERR# when it is unable to deliver Posted Write data after 2 <sup>24</sup> attempts [or programmed Maximum Retry count (TOCNTRL[2:0]; PCI:45h Shadow register)]. S_SERR# is asserted if this event occurs when bit is 0 and the Command register SERR# Enable bit is set (PCISCR[8]=1; Primary PCI:44h, Secondary PCI:04h).	Yes	Yes	0
3	<b>Target Abort during Posted Write.</b> Controls PCI 6540 ability to assert S_SERR# when it receives a Target Abort while attempting to deliver Posted Write data. S_SERR# is asserted if this event occurs when bit is 0 and the Command register P_SERR# Enable bit is set (PCICR[8]=1; Primary PCI:04h, Secondary PCI:44h).	Yes	Yes	0
4	<b>Master Abort on Posted Write.</b> Controls PCI 6540 ability to assert S_SERR# when it receives a Master Abort while attempting to deliver Posted Write data. S_SERR# is asserted if this event occurs when bit is 0 and Command register P_SERR# Enable bit is set (PCICR[8]=1; Primary PCI:04h, Secondary PCI:44h).	Yes	Yes	0
5	<b>Delayed Configuration or I/O Write Non-Delivery.</b> Controls PCI 6540 ability to assert S_SERR# when it is unable to deliver Delayed Write data after 2 <sup>24</sup> attempts [or programmed Maximum Retry count (TOCNTRL[2:0]; PCI:45h Shadow register)]. S_SERR# is asserted if this event occurs when bit is 0 and the Command register SERR# Enable bit is set (PCICR[8]=1; Primary PCI:04h, Secondary PCI:44h).	Yes	Yes	0
6	<b>Delayed Read-No Data from Target.</b> Controls PCI 6540 ability to assert S_SERR# when it is unable to transfer Read data from the target after 2 <sup>24</sup> attempts [or programmed Maximum Retry count (TOCNTRL[2:0]; PCI:45h Shadow register)]. S_SERR# is asserted if this event occurs when bit is 0 and the Command register SERR# Enable bit is set (PCICR[8]=1; Primary PCI:04h, Secondary PCI:44h).	Yes	Yes	0
7	<b>Posted Write Data Parity Error.</b> S_SERR# is asserted because a Posted Write Data Parity error occurred on the target bus.	Yes	Yes/Clr	0

Register 6-177. (PSSERRSR; PCI:98h) P\_SERR# and S\_SERR# Status

Bit	Description	Read	Write	Value after Reset
0	<b>Primary Post Write Non-Delivery.</b> P_SERR# is asserted because PCI 6540 was unable to deliver Posted Write data to the target before the Timeout Counter expired.	Yes	Yes/Clr	0
1	<b>Primary Delayed Write Non-Delivery.</b> P_SERR# is asserted because PCI 6540 was unable to deliver Delayed Write data before the Timeout Counter expired.	Yes	Yes/Clr	0
2	<b>Primary Delayed Read Failed.</b> P_SERR# is asserted because PCI 6540 was unable to read data from the target before the Timeout Counter expired.	Yes	Yes/Clr	0
3	<b>Primary Transaction Master Timeout.</b> P_SERR# is asserted because a master did not repeat a Read or Write transaction before the initiator bus Master Timeout Counter expired.	Yes	Yes/Clr	0
4	<b>Secondary Post Write Non-Delivery.</b> S_SERR# is asserted because PCI 6540 was unable to deliver Posted Write data to the target before the Timeout Counter expired.	Yes	Yes/Clr	0
5	<b>Secondary Delayed Write Non-Delivery.</b> S_SERR# is asserted because PCI 6540 was unable to deliver Delayed Write data before the Timeout Counter expired.	Yes	Yes/Clr	0
6	<b>Secondary Delayed Read Failed.</b> S_SERR# is asserted because PCI 6540 was unable to read data from the target before the Timeout Counter expired.	Yes	Yes/Clr	0
7	<b>Secondary Transaction Master Timeout.</b> S_SERR# is asserted because a master did not repeat a Read or Write transaction before the initiator bus Master Timeout Counter expired.	Yes	Yes/Clr	0

## 6.2.4.5 GPIO[3:0] Registers

Register 6-178. (GPIOOD[3:0]; PCI:99h) GPIO[3:0] Output Data

Bit	Description	Read	Write	Value after Reset
3:0	<b>GPIO[3:0] Output Data Write 1 to Clear.</b> Writing 1 to any of these bits drives the corresponding signal low on the GPIO[3:0] bus if the signal is programmed as an output. Writing 0 has no effect. Read returns the last written value.	Yes	Yes/Clr	0h
7:4	<b>GPIO[3:0] Output Data Write 1 to Set.</b> Writing 1 to any of these bits drives the corresponding signal high on the GPIO[3:0] bus if the signal is programmed as an output. Writing 0 has no effect. Read returns the last written value.	Yes	Yes/Set High	0h

Register 6-179. (GPIOOE[3:0]; PCI:9Ah) GPIO[3:0] Output Enable

Bit	Description	Read	Write	Value after Reset
3:0	<b>GPIO[3:0] Output Enable Write 1 to Clear.</b> Writing 1 to any of these bits configures the corresponding signal on the GPIO[3:0] bus as an input. Writing 0 has no effect. Read returns the last written value.	Yes	Yes/Clr	0h
7:4	<b>GPIO[3:0] Output Enable Write 1 to Set.</b> Writing 1 to any of these bits configures the corresponding signal on the GPIO[3:0] bus as an output. Writing 0 has no effect. Read returns the last written value.	Yes	Yes/Set High	0h

Register 6-180. (GPIOID[3:0]; PCI:9Bh) GPIO[3:0] Input Data

Bit	Description	Read	Write	Value after Reset
3:0	<b>Reserved.</b>	Yes	No	0h
7:4	<b>GPIO[3:0] Input Data.</b> Reads GPIO[3:0] pin state. The state is updated on the PCI Clock cycle following a change in the GPIO[3:0] state.	Yes	No	—



## 6.2.4.6 Hot Swap and Read-Only Control Register

Register 6-181. (HSSRRC; PCI:9Ch) Hot Swap Switch and ROR Control

Bit	Description	Read	Write	Value after Reset
0	<b>Hot Swap Extraction Switch.</b> Used to signal board extraction. If set, the board is in the inserted state. Writing 0 to this bit signals pending board extraction.	Yes	Yes	—
1	<b>Primary Port 64-Bit Extension Signals Park.</b> Value: 1 = PCI 6540 drives primary port PCI 64-bit extension signals P_AD[63:32], P_CBE[7:4]#, and P_PAR64 to 0	Yes	Yes	0
2	<b>Secondary Port 64-Bit Extension Signals Park.</b> Value: 1 = PCI 6540 drives secondary port PCI 64-bit extension signals S_AD[63:32], S_CBE[7:4]#, and S_PAR64 to 0	Yes	Yes	0
6:3	<b>Reserved.</b>	Yes	No	0h
7	<p><b>Read-Only Registers Write Enable.</b> Setting this bit to 1 enables writes to specific bits within these normally Read-Only registers (refer to the listed registers for further details):</p> <ul style="list-style-type: none"> <li>• Vendor and Device IDs (PCIIDR; Primary PCI:00h, Secondary PCI:40h and Primary PCI:40h, Secondary PCI:00h)</li> <li>• PCI Class Code (PCICCR; Primary PCI:09h – 0Bh, Secondary PCI:49h – 4Bh and Primary PCI:49h – 4Bh, Secondary PCI:09h – 0Bh)</li> <li>• PCI Header Type (PCIHTR; Primary PCI:0Eh, Secondary PCI:4Eh and Primary PCI:4Eh, Secondary PCI:0Eh)</li> <li>• Subsystem Vendor ID (PCISVID; Primary PCI:2Ch, Secondary PCI:6Ch and Primary PCI:6Ch, Secondary PCI:2Ch)</li> <li>• Subsystem ID (PCISID; Primary PCI:2Eh, Secondary PCI:6Eh and Primary PCI:6Eh, Secondary PCI:2Eh)</li> <li>• Primary Minimum Grant (PCIPMGR; PCI:3Eh, Secondary PCI:7Eh)</li> <li>• Primary Maximum Latency (PCIPMLR; PCI:3Fh, Secondary PCI:7Fh)</li> <li>• Secondary Minimum Grant (PCISMGR; PCI:7Eh, Secondary PCI:3Eh)</li> <li>• Secondary Maximum Latency (PCISMLR; PCI:7Fh, Secondary PCI:3Fh)</li> <li>• Message Signaled Interrupts Capability ID (MSICAPID; PCI:AC Shadow register)</li> <li>• Power Management Capabilities (PMC; PCI:DE Shadow register)</li> <li>• Power Management Control/Status (PMCSR; PCI:E0 Shadow register)</li> <li>• Power Management Data (PMCDATA; PCI:E3 Shadow register)</li> </ul> <p>Bit must be cleared after the values are modified in these Read-Only registers.</p>	Yes	Yes	0

## 6.2.4.7 GPIO[7:4], Power-Up Status, and GPIO[15:8] Registers

**Register 6-182. (GPIOOD[7:4]; PCI:9Dh) GPIO[7:4] Output Data**

Bit	Description	Read	Write	Value after Reset
3:0	<b>GPIO[7:4] Output Data Write 1 to Clear.</b> Writing 1 to any of these bits drives the corresponding signal low on the GPIO[7:4] bus if the signal is programmed as an output. Writing 0 has no effect. Read returns the last written value.	Yes	Yes/Clr	0h
7:4	<b>GPIO[7:4] Output Data Write 1 to Set.</b> Writing 1 to any of these bits drives the corresponding signal high on the GPIO[7:4] bus if the signal is programmed as an output. Writing 0 has no effect. Read returns the last written value.	Yes	Yes/Set High	0h

**Register 6-183. (GPIOOE[7:4]; PCI:9Eh) GPIO[7:4] Output Enable**

Bit	Description	Read	Write	Value after Reset
3:0	<b>GPIO[7:4] Output Enable Write 1 to Clear.</b> Writing 1 to any of these bits configures the corresponding signal on the GPIO[7:4] bus as an input. Writing 0 has no effect. Read returns the last written value.	Yes	Yes/Clr	0h
7:4	<b>GPIO[7:4] Output Enable Write 1 to Set.</b> Writing 1 to any of these bits configures the corresponding signal on the GPIO[7:4] bus as an output. Writing 0 has no effect. Read returns the last written value.	Yes	Yes/Set High	0h

**Register 6-184. (GPIOID[7:4]; PCI:9Fh) GPIO[7:4] Input Data**

Bit	Description	Read	Write	Value after Reset
3:0	<i>Reserved.</i>	Yes	No	0h
7:4	<b>GPIO[7:4] Input Data.</b> Reads the GPIO[7:4] pins state. The state is updated on the PCI Clock cycle following a change in the GPIO[7:4] state.	Yes	No	—

**Register 6-185. (PWRUPSR; PCI:A0h) Power-Up Status**

Bit	Description	Read	Write	Value after Reset
7:0	<p><b>Power-Up Latched Status Bits.</b> Upon PWRGD (power good), the status of GPIO[15:8] are latched into PWRUPSR. Select pin status for desired option setting or checking. Recommended use:</p> <ul style="list-style-type: none"> <li>• <b>GPIO15—Primary Power State.</b> Value of 1h indicates primary port power is stable.</li> <li>• <b>GPIO14—Secondary Power State.</b> Value of 1h indicates secondary port power is stable.</li> <li>• <b>GPIO13—100 MHz PCI-X.</b> Value of 1h indicates secondary PCI-X Bus is running at 100 MHz.</li> </ul>	Yes	No	GPIO[15:8]

**Register 6-186. (GPIOOD[15:8]; PCI:A1h) GPIO[15:8] Output Data**

Bit	Description	Read	Write	Value after Reset
7:0	<p><b>GPIO[15:8] Output Data.</b> Values written to this register are output on the GPIO[15:8] pins, if enabled. Values: 0h = Low 1h = High</p>	Yes	Yes	0h

**Register 6-187. (GPIOOE[15:8]; PCI:A2h) GPIO[15:8] Output Enable**

Bit	Description	Read	Write	Value after Reset
7:0	<p><b>GPIO[15:8] Output.</b> Writing 1 to any of these bits configures the corresponding signal on the GPIO[15:8] bus as an output. Writing 0 configures the corresponding signal as an input.</p>	Yes	Yes	0h

**Register 6-188. (GPIOID[15:8]; PCI:A3h) GPIO[15:8] Input Data**

Bit	Description	Read	Write	Value after Reset
7:0	<p><b>GPIO[15:8] Input Data.</b> Reads the GPIO[15:8] pins state. The state is updated on the PCI Clock cycle following a change in the GPIO[15:8] state.</p>	Yes	No	—

### 6.2.4.8 Direct Message Interrupt Registers

When enabled, a write command to the following message registers can cause a PCI interrupt. The Direct Message Interrupts encode the interrupt message in the response and are therefore faster than using the doorbell registers. S\_INTA# is activated by downstream messages, and P\_INTA# is activated by upstream messages.

**Register 6-189. (UPSMSG0; PCI:A4h) Upstream Message 0**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Upstream Message 0.</b> Secondary port masters can write data to UPSMSG0 for primary port devices to read.	Yes	Yes	0h

**Register 6-190. (UPSMSG1; PCI:A5h) Upstream Message 1**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Upstream Message 1.</b> Secondary port masters can write data to UPSMSG1 for primary port devices to read.	Yes	Yes	0h

**Register 6-191. (UPSMSG2; PCI:A6h) Upstream Message 2**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Upstream Message 2.</b> Secondary port masters can write data to UPSMSG2 for primary port devices to read.	Yes	Yes	0h

**Register 6-192. (UPSMSG3; PCI:A7h) Upstream Message 3**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Upstream Message 3.</b> Secondary port masters can write data to UPSMSG3 for primary port devices to read.	Yes	Yes	0h

**Register 6-193. (DWNMSG0; PCI:A8h) Downstream Message 0**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Downstream Message 0.</b> Primary port masters can Write data to DWNMSG0 for secondary port devices to read. When this is written, Downstream Interrupt Status bit is set (DWNINTSR[0]=1; PCI:CAh) and S_INTA# is asserted. The secondary device reads DWNINTSR[0] and writes 1 to clear it and de-assert S_INTA#.	Yes	Yes	0h

**Register 6-194. (DWNMSG1; PCI:A9h) Downstream Message 1**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Downstream Message 1.</b> Primary port masters can Write data to DWNMSG1 for secondary port devices to read. When this is written, Downstream Interrupt Status bit is set (DWNINTSR[1]=1; PCI:CAh) and S_INTA# is asserted. The secondary device reads DWNINTSR[1] and writes 1 to clear it and de-assert S_INTA#.	Yes	Yes	0h

**Register 6-195. (DWNMSG2; PCI:AAh) Downstream Message 2**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Downstream Message 2.</b> Primary port masters can Write data to DWNMSG2 for secondary port devices to read. When this is written, Downstream Interrupt Status bit is set (DWNINTSR[2]=1; PCI:CAh) and S_INTA# is asserted. The secondary device reads DWNINTSR[2] and writes 1 to clear it and de-assert S_INTA#.	Yes	Yes	0h

**Register 6-196. (DWNMSG3; PCI:ABh) Downstream Message 3**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Downstream Message 3.</b> Primary port masters can Write data to DWNMSG3 for secondary port devices to read. When this is written, Downstream Interrupt Status bit is set (DWNINTSR[3]=1; PCI:CAh) and S_INTA# is asserted. The secondary device reads DWNINTSR[3] and writes 1 to clear it and de-assert S_INTA#.	Yes	Yes	0h

## 6.2.4.9 Message Signaled Interrupt Registers

**Register 6-197. (MSICAPID; PCI:ACH) Message Signaled Interrupts Capability ID**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Message Signaled Interrupts (MSI) Capability ID.</b> PCI-SIG-issued Capability ID for Message Signaled Interrupts is 05h.	Yes	Only if HSSRRRC[7]=1; Serial EEPROM	05h

**Register 6-198. (MSINEXT; PCI:ADh) Message Signaled Interrupts Next Capability Pointer**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Message Signaled Interrupt Next Cap Pointer.</b> Offset into PCI Configuration space for location of the next capability in the New Capabilities Linked List. Set to 0h, as this is the last item in the linked list.	Yes	No	0h

**Register 6-199. (MSIC; PCI:AEh) Message Signaled Interrupts Control**

Bit	Description	Read	Write	Value after Reset
0	<b>Message Signaled Interrupts Enable.</b> Set by system configuration software to enable MSI.	Yes	Yes	0
3:1	<b>Multiple Message Capable.</b> System configuration software reads these bits to determine the number of requested messages.	Yes	No	000b
6:4	<b>Multiple Message Enable.</b> System configuration software writes to these bits to indicate the number of allocated messages.	Yes	Yes	000b
7	<b>64-Bit Address Capable.</b> System configuration software reads this bit to determine whether PCI 6540 uses 64-bit addressing. Values: 0 = 32 bit 1 = 64 bit	Yes	No	0
15:8	<b>Reserved.</b>	Yes	No	0h

**Register 6-200. (MSIADDR; PCI:B0h) Message Signaled Interrupts Address**

Bit	Description	Read	Write	Value after Reset
1:0	<i>Reserved.</i>	Yes	No	00b
31:2	<b>Message Signaled Interrupt Address.</b> System-specified message address.	Yes	Yes	0h

**Register 6-201. (MSIUADDR; PCI:B4h) Message Signaled Interrupts Upper Address**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Message Signaled Interrupt Upper Address.</b> System-specified message upper address.	Yes	Yes	0h

**Register 6-202. (MSIDATA; PCI:B8h) Message Signaled Interrupts Data**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Message Signaled Interrupt Data.</b> System-specified message. Each MSI function is allocated up to 32 unique messages.	Yes	Yes	0h
31:16	<i>Reserved.</i>	Yes	No	0h

## 6.2.4.10 Doorbell and Miscellaneous Interrupt Registers

S\_INTA# is asserted when there are active downstream interrupt sources. P\_INTA# is asserted when there are active upstream interrupt sources.

**Register 6-203. (DWNDBIE; PCI:C0h) Downstream Doorbell Interrupt Enable**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Secondary Interrupt Requests Enable.</b> Set to 1 to enable appropriate Downstream Doorbell Interrupt Request bit (DWNDBIR[15:0]; PCI:C2h) to the secondary port.	Yes	Yes	0h

**Register 6-204. (DWNDBIR; PCI:C2h) Downstream Doorbell Interrupt Request**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Downstream Doorbell Interrupt Request.</b> If a primary master sets any of these bits to 1, causes secondary port interrupts. When one of these bits is 1, the corresponding Doorbell Interrupt Status bit (DWNDBIS[15:0]; PCI:C8h) cannot be cleared and new interrupts are generated. Therefore, immediately clear these bits after they are set to generate an interrupt.	Yes	Yes	0h

**Register 6-205. (UPSDBIE; PCI:C4h) Upstream Doorbell Interrupt Enable**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Primary Interrupt Requests Enable.</b> Set to 1 to enable appropriate software interrupt request bit (UPSDBIR[15:0]; PCI:C6h) to the primary port.	Yes	Yes	0h

**Register 6-206. (UPSDBIR; PCI:C6h) Upstream Doorbell Interrupt Request**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Upstream Doorbell Interrupt Request.</b> If a secondary master sets any of these bits to 1, causes Primary Port interrupts. When one of these bits is 1, the corresponding Doorbell Interrupt Status bit (UPSDBIS[15:0]; PCI:CCh) cannot be cleared and new interrupts are generated. Therefore, immediately clear these bits after they are set to generate an interrupt.	Yes	Yes	0h



**Register 6-207. (DWNDBIS; PCI:C8h) Downstream Doorbell Interrupt Status**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Secondary Interrupt Requests Status.</b> Set bit indicates the corresponding secondary Software Interrupt Request (DWNDBIR; PCI:C2h) to the secondary host was detected.	Yes	Yes/Clr	0h

**Register 6-208. (DWNINTSR; PCI:CAh) Downstream Interrupt Status**

Bit	Description	Read	Write	Value after Reset
0	<b>Downstream Message 0.</b> Primary-to-secondary Message 0 is written. Write 1 to de-assert S_INTA#.	Yes	Yes/Clr	0
1	<b>Downstream Message 1.</b> Primary-to-secondary Message 1 is written. Write 1 to de-assert S_INTA#.	Yes	Yes/Clr	0
2	<b>Downstream Message 2.</b> Primary-to-secondary Message 2 is written. Write 1 to de-assert S_INTA#.	Yes	Yes/Clr	0
3	<b>Downstream Message 3.</b> Primary-to-secondary Message 3 is written. Write 1 to de-assert S_INTA#.	Yes	Yes/Clr	0
4	<b>P_RSTIN# De-Assertion.</b> P_RSTIN# de-assertion detected.	Yes	Yes/Clr	0
5	<b>P_PME# De-Assertion.</b> P_PME# de-assertion detected.	Yes	Yes/Clr	0
6	<b>GPIO15 Active Low Interrupt.</b> Recommended use: Primary power is not available. Reflects GPIO15 pin inverted state, if this interrupt is enabled; otherwise, value is 0.	Yes	No	0
7	<b>GPIO5 Active Low Interrupt.</b> Reflects GPIO5 pin inverted state, if this interrupt is enabled; otherwise, value is 0.	Yes	No	0

**Register 6-209. (UPSINTE; PCI:CBh) Upstream Interrupt Enable**

Bit	Description	Read	Write	Value after Reset
0	<b>Upstream Message 0 Interrupt Enable.</b> Enables secondary-to-primary Message 0 event interrupt trigger.	Yes	Yes	0
1	<b>Upstream Message 1 Interrupt Enable.</b> Enables secondary-to-primary Message 1 event interrupt trigger.	Yes	Yes	0
2	<b>Upstream Message 2 Interrupt Enable.</b> Enables secondary-to-primary Message 2 event interrupt trigger.	Yes	Yes	0
3	<b>Upstream Message 3 Interrupt Enable.</b> Enables secondary-to-primary Message 3 event interrupt trigger.	Yes	Yes	0
4	<b>S_RSTIN# De-Assertion Enable.</b> Enables S_RSTIN# de-assertion detection.	Yes	Yes	0
5	<b>S_PME# De-Assertion Enable.</b> Enables S_PME# de-assertion detection.	Yes	Yes	0
6	<b>Secondary External Interrupt at GPIO14 Pin.</b> Enables interrupt trigger when GPIO14 pin is low.	Yes	Yes	0
7	<b>Secondary External Interrupt at GPIO4 Pin.</b> Enables interrupt trigger when GPIO4 pin is low.	Yes	Yes	0

**Register 6-210. (UPSDIB; PCI:CCh) Upstream Doorbell Interrupt Status**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Primary Interrupt Requests Status.</b> Set bit indicates the corresponding Primary Software Interrupt request UPSDBIR; PCI:C6h) to the primary host was detected.	Yes	Yes/Clr	0h

**Register 6-211. (UPSINTSR; PCI:CEh) Upstream Interrupt Status**

Bit	Description	Read	Write	Value after Reset
0	<b>Upstream Message 0.</b> Secondary-to-primary Message 0 is written. Write 1 to de-assert P_INTA#.	Yes	Yes/Clr	0
1	<b>Upstream Message 1.</b> Secondary-to-primary Message 1 is written. Write 1 to de-assert P_INTA#.	Yes	Yes/Clr	0
2	<b>Upstream Message 2.</b> Secondary-to-primary Message 2 is written. Write 1 to de-assert P_INTA#.	Yes	Yes/Clr	0
3	<b>Upstream Message 3.</b> Secondary-to-primary Message 3 is written. Write 1 to de-assert P_INTA#.	Yes	Yes/Clr	0
4	<b>S_RSTIN# De-Assertion.</b> S_RSTIN# de-assertion detected.	Yes	Yes/Clr	0
5	<b>S_PME# De-Assertion.</b> S_PME# de-assertion detected.	Yes	Yes/Clr	0
6	<b>GPIO14 Active Low Interrupt.</b> Recommended use: Secondary power is not available. Reflects GPIO14 pin inverted state, if this interrupt is enabled; otherwise, value is 0.	Yes	No	0
7	<b>GPIO4 Active Low Interrupt.</b> Reflects GPIO4 pin inverted state, if this interrupt is enabled; otherwise, value is 0.	Yes	No	0

**Register 6-212. (DWNINTE; PCI:CFh) Downstream Interrupt Enable**

Bit	Description	Read	Write	Value after Reset
0	<b>Downstream Message 0 Interrupt Enable.</b> Enables primary-to- secondary Message 0 event interrupt trigger.	Yes	Yes	0
1	<b>Downstream Message 1 Interrupt Enable.</b> Enables primary-to- secondary Message 1 event interrupt trigger.	Yes	Yes	0
2	<b>Downstream Message 2 Interrupt Enable.</b> Enables primary-to- secondary Message 2 event interrupt trigger.	Yes	Yes	0
3	<b>Downstream Message 3 Interrupt Enable.</b> Enables primary-to- secondary Message 3 event interrupt trigger.	Yes	Yes	0
4	<b>P_RSTIN# De-Assertion Enable.</b> Enables P_RSTIN# de-assertion detection.	Yes	Yes	0
5	<b>P_PME# De-Assertion Enable.</b> Enables P_PME# de-assertion detection.	Yes	Yes	0
6	<b>Primary External Interrupt at GPIO15 Pin.</b> Enables interrupt trigger when GPIO15 pin is low.	Yes	Yes	0
7	<b>Primary External Interrupt at GPIO5 Pin.</b> Enables interrupt trigger when GPIO5 pin is low.	Yes	Yes	0

### 6.2.4.11 Non-Transparent Configuration Ownership Semaphore Register

**Register 6-213. (NTCOS; PCI:D2h) Non-Transparent Configuration Ownership Semaphore**

Bit	Description	Read	Write	Value after Reset
0	<b>Non-Transparent Configuration Ownership Semaphore Mechanism.</b> When the primary or secondary port performs a Configuration Read to this bit, it returns 0 if there is no Configuration Read beforehand. Such a read automatically sets this bit to 1. Further reads by other primary or secondary masters observe a 1 (previously owned). Bit must be cleared by the master that set the bit, using a Configuration Write 1 to NTCOS. Software can check this semaphore mechanism status by way of CCNTRL[0]; PCI:D8h without taking ownership.	Yes	Yes/Clr	0
7:1	<i>Reserved.</i>	Yes	No	0h

### 6.2.4.12 Sticky Scratch and Extended Registers

There are eight, 32-bit Sticky Scratch registers available in the PCI 6540 (SCRATCHx; EXT:00h to 07h). Address translation registers are also located in the Extended Register Area. The extended registers

are accessed by way of the Extended Register Index and Extended Register Data registers (EXTRIDX; PCI:D3h and EXTRDATA; PCI:D4h, respectively).

The Base Address registers at index 08h to 0Fh are discussed in Section 6.2.4.13.

**Table 6-8. Extended Register Map—Offset from Extended Register Index, Non-Transparent Mode**

Extended Register Index	31	24	23	16	15	8	7	0	Writable	Serial EEPROM Writable
00h	32-Bit Sticky 0								Yes	No
01h	32-Bit Sticky 1								Yes	No
02h	32-Bit Sticky 2								Yes	No
03h	32-Bit Sticky 3								Yes	No
04h	32-Bit Sticky 4								Yes	No
05h	32-Bit Sticky 5								Yes	No
06h	32-Bit Sticky 6								Yes	No
07h	32-Bit Sticky 7								Yes	No

**Note:** When the serial EEPROM is set to initialize for Universal Non-Transparent mode applications, these registers also activate translation in Universal Transparent mode if PRV\_DEV=1. (Refer to Section 7, "Serial EEPROM.")

**Register 6-214. (SCRATCHx; Ext:00h – 07h) 32-Bit Sticky Scratch**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Sticky Scratch.</b> Upon Power Good, the values of these registers are undefined. After Power is Good, P_RSTIN# and/or S_RSTIN# assertion does not affect their current value.	Yes	Yes	—

**Register 6-215. (EXTRIDX; PCI:D3h) Extended Register Index**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Extended Index Address.</b> Extended registers index address.	Yes	Yes	—

**Register 6-216. (EXTRDATA; PCI:D4h) Extended Register Data**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Extended Register Data.</b> Configuration Write causes the data presented at this port to be written into the register addressed by the Extended Register Index (EXTRIDX; PCI:D3h). Configuration Read causes the data from the register addressed by the Extended Register Index (EXTRIDX; PCI:D3h) to be placed into and read from EXTRDATA.	Yes	Yes	—

### 6.2.4.13 Address Translation Control Registers

When using the PCI 6540 in standard Non-Transparent mode, the Address Translation Control registers should remain in the default state. These registers are used only when address translation is required in Non-Transparent mode. (Refer to Section 10.7.3, “Non-Transparent Mode Address Translation,” for further details.)

The Address Translation Enable Control bits enable or disable only the Address Translation functions. These bits do not control whether the Memory window is open. Take care to ensure that there is a valid Memory window in the memory map of the host on that port.

These registers are accessible only by reading/writing through the Extended Register Index and Data registers (EXTRIDX; PCI:D3h and EXTRDATA; PCI:D4h, respectively).

**Table 6-9. Extended Register Map (Used in Non-Transparent Address Translation)—Offset from Extended Register Index**

Extended Register Index	3124231615870								Writable	Serial EEPROM Writable
08h	Upstream BAR 0 Translation Address								Yes	Yes
09h	Upstream BAR 1 Translation Address								Yes	Yes
0Ah	Upstream BAR 2 or Upstream BAR 1 Upper 32 Bits Translation Address								Yes	Yes
0Bh	Upstream Translation Enable	Upstream BAR 2 Translation Mask		Upstream BAR 1 Translation Mask		Upstream BAR 0 Translation Mask		Yes	Yes	
0Ch	Downstream BAR 0 Translation Address								Yes	Yes
0Dh	Downstream BAR 1 Translation Address								Yes	Yes
0Eh	Downstream BAR 2 or Downstream BAR 1 Upper 32 Bits Translation Address								Yes	Yes
0Fh	Downstream Translation Enable	Downstream BAR 2 Translation Mask		Downstream BAR 1 Translation Mask		Downstream BAR 0 Translation Mask		Yes	Yes	

**Note:** When the serial EEPROM is set to initialize for Universal Non-Transparent mode applications, these registers also activate translation in Universal Transparent mode if PRV\_DEV=1. (Refer to Section 10.7.2, “Transparent Mode Address Translation,” on page 10-6 and Section 10.7.3, “Non-Transparent Mode Address Translation,” on page 10-11 for further details.)

**Register 6-217. (UPSTNBAR0; Ext:08h) Upstream BAR 0 Translation Address**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Upstream BAR 0 Translation Address.</b> Bits [11:0] are Read-Only, and always 0. Only Address bits [31:12] are translated. Lower Address bits are passed. <b>Note:</b> Translation address must align with the Window Size boundary.	Yes	Yes [31:12]; Serial EEPROM	0h

**Register 6-218. (UPSTNBAR1; Ext:09h) Upstream BAR 1 Translation Address**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Upstream BAR 1 Translation Address.</b> Bits [19:0] are Read-Only, and always 0. Only Address bits [31:20] are translated. Lower Address bits are passed. <b>Note:</b> Translation address must align with the Window Size boundary.	Yes	Yes [31:20]; Serial EEPROM	0h

**Register 6-219. (UPSTNBAR2; Ext:0Ah) Upstream BAR 2 Translation Address or Upstream BAR 1 Upper 32 Bits**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Upstream BAR 2 Translation Address or Upstream BAR 1 Upper 32 Bits.</b> Bits [11:0] are Read-Only, and always 0. Only Address bits [31:12] are translated. Lower Address bits are passed. If PCIUBAR1 is configured as a 64-bit BAR (UPSBAR1MSK[14]=1; EXT:0Bh), then UPSTNBAR2 contains the upper 32 bits of the BAR 1 Translation address. <b>Note:</b> Translation address must align with the Window Size boundary.	Yes	Yes [31:12]; Serial EEPROM	0h

**Register 6-220. (UPSBAR0MSK; Ext:0Bh) Upstream BAR 0 Translation Mask**

Bit	Description	Read	Write	Value after Reset
4:0	<b>Address Mask MSB Position.</b> Number of Local Address bits for BAR 0 mask.	Yes	Yes; Serial EEPROM	1Fh
5	<b>Reserved.</b>	Yes	No	0
6	<b>BAR Type.</b> Values: 0 = BAR 0 points to I/O space 1 = BAR 0 points to Memory space	Yes	Yes; Serial EEPROM	0
7	<b>Prefetchable.</b> Values: 0 = Region pointed to by BAR 0 is not prefetchable 1 = Region pointed to by BAR 0 is in a Prefetchable Memory region	Yes	Yes; Serial EEPROM	0

**Register 6-221. (UPSBAR1MSK; Ext:0Bh) Upstream BAR 1 Translation Mask**

Bit	Description	Read	Write	Value after Reset
13:8	<b>Address Mask MSB Position.</b> Number of Local Address bits for BAR 1 mask.	Yes	Yes; Serial EEPROM	3Fh
14	<b>BAR Type.</b> Values: 0 = BAR 1 is a 32-bit BAR 1 = BAR 1 is a 64-bit BAR	Yes	Yes; Serial EEPROM	0
15	<b>Prefetchable.</b> Values: 0 = Region pointed to by BAR 1 is not prefetchable 1 = Region pointed to by BAR 1 is in a Prefetchable Memory region	Yes	Yes; Serial EEPROM	0

**Register 6-222. (UPSBAR2MSK; Ext:0Bh) Upstream BAR 2 Translation Mask**

Bit	Description	Read	Write	Value after Reset
20:16	<b>Address Mask MSB Position.</b> Number of Local Address bits for BAR 2 mask.	Yes	Yes; Serial EEPROM	1Fh
22:21	<b>Reserved.</b>	Yes	No	00b
23	<b>Prefetchable.</b> Values: 0 = Region pointed to by BAR 2 is not prefetchable 1 = Region pointed to by BAR 2 is in a Prefetchable Memory region	Yes	Yes; Serial EEPROM	0



Register 6-223. (UPSTNE; Ext:0Bh) Upstream Translation Enable

Bit	Description	Read	Write	Value after Reset
24	<b>Upstream BAR 0 Enable.</b> If set to 1, address translation using BAR 0 is enabled.	Yes	Yes; Serial EEPROM	0
25	<b>Upstream BAR 1 Enable.</b> If set to 1, address translation using BAR 1 is enabled.	Yes	Yes; Serial EEPROM	0
26	<b>Upstream BAR 2 Enable.</b> If set to 1, address translation using BAR 2 is enabled.	Yes	Yes; Serial EEPROM	0
30:27	<b>Reserved.</b>	Yes	No	0h
31	<b>S_PORT_READY.</b> Upon S_RSTIN# assertion, bit is cleared. Set by the secondary port master upon completion of secondary port initialization. When P_BOOT=0 (secondary port retains boot priority), the primary port master access to PCI Standard BAR configurations at offsets 10h to 1Bh is Retried until the S_PORT_READY bit is set. When P_BOOT=0 and bit is 0, cross-bridge traffic initiated by primary port is returned with Retry. S_PORT_READY mechanism does not have the above effect if the special fixed-size cross-bridge communication window is enabled by setting the XB_MEM input to 1.	Yes	Yes; Serial EEPROM	—

**Register 6-224. (DWNTNBAR0; Ext:0Ch) Downstream BAR 0 Translation Address**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Downstream BAR 0 Translation Address.</b> Bits [11:0] are Read-Only, and always 0. Only Address bits [31:12] are translated. Lower Address bits are passed. <b>Note:</b> Translation address must align with the Window Size boundary.	Yes	Yes [31:12]; Serial EEPROM	0h

**Register 6-225. (DWNTNBAR1; Ext:0Dh) Downstream BAR 1 Translation Address**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Downstream BAR 1 Translation Address.</b> Bits [19:0] are Read-Only, and always 0. Only Address bits [31:20] are translated. Lower Address bits are passed. <b>Note:</b> Translation address must align with the Window Size boundary.	Yes	Yes [31:20]; Serial EEPROM	0h

**Register 6-226. (DWNTNBAR2; Ext:0Eh) Downstream BAR 2 or Downstream Memory BAR 1 Upper 32 Bits Translation Address**

Bit	Description	Read	Write	Value after Reset
31:0	<b>Downstream BAR 2 or Downstream Memory BAR 1 Upper 32 Bits Translation Address.</b> Bits [11:0] are Read-Only and always 0. Only Address bits [31:12] are translated. Lower Address bits are passed. If PCIBAR1 is configured as a 64-bit BAR (DWNBAR1MSK[14]=1; Ext:0Fh), then DWNTNBAR2 contains the upper 32 bits of the BAR 1 Translation address. <b>Note:</b> Translation address must align with the Window Size boundary.	Yes	Yes [31:12]; Serial EEPROM	0h

**Register 6-227. (DWNBAR0MSK; Ext:0Fh) Downstream BAR 0 Translation Mask**

Bit	Description	Read	Write	Value after Reset
4:0	<b>Address Mask MSB Position.</b> Number of Local Address bits for BAR 0 mask.	Yes	Yes; Serial EEPROM	1Fh
5	<b>Reserved.</b>	Yes	No	0
6	<b>BAR Type.</b> Values: 0 = BAR 0 points to Memory space 1 = BAR 0 points to I/O space	Yes	Yes; Serial EEPROM	0
7	<b>Prefetchable.</b> Values: 0 = Region pointed to by BAR 0 is not prefetchable 1 = Region pointed to by BAR 0 is in a Prefetchable Memory region	Yes	Yes; Serial EEPROM	0

**Register 6-228. (DWNBAR1MSK; Ext:0Fh) Downstream BAR 1 Translation Mask**

Bit	Description	Read	Write	Value after Reset
13:8	<b>Address Map MSB Position.</b> Number of Local Address bits for BAR 1 mask.	Yes	Yes; Serial EEPROM	3Fh
14	<b>BAR Type.</b> Values: 0 = BAR 1 is a 32-bit BAR 1 = BAR 1 is a 64-bit BAR	Yes	Yes; Serial EEPROM	0
15	<b>Prefetchable.</b> Values: 0 = Region pointed to by BAR 1 is not prefetchable 1 = Region pointed to by BAR 1 is in a Prefetchable Memory region	Yes	Yes; Serial EEPROM	0

**Register 6-229. (DWNBAR2MSK; Ext:0Fh) Downstream BAR 2 Translation Mask**

Bit	Description	Read	Write	Value after Reset
20:16	<b>Address Mask MSB Position.</b> Number of Local Address bits for BAR 2 mask.	Yes	Yes; Serial EEPROM	1Fh
22:21	<b>Reserved.</b>	Yes	No	00b
23	<b>Prefetchable.</b> Values: 0 = Region pointed to by BAR 2 is not prefetchable 1 = Region pointed to by BAR 2 is in a Prefetchable Memory region	Yes	Yes; Serial EEPROM	0

Register 6-230. (DWNTNE; Ext:0Fh) Downstream Translation Enable

Bit	Description	Read	Write	Value after Reset
24	<b>Downstream BAR 0 Enable.</b> If set to 1, address translation using BAR 0 is enabled.	Yes	Yes; Serial EEPROM	0
25	<b>Downstream BAR 1 Enable.</b> If set to 1, address translation using BAR 1 is enabled.	Yes	Yes; Serial EEPROM	0
26	<b>Downstream BAR 2 Enable.</b> If set to 1, address translation using BAR 2 is enabled.	Yes	Yes; Serial EEPROM	0
30:27	<b>Reserved.</b>	Yes	No	0h
31	<p><b>P_PORT_READY.</b> Upon P_RSTIN# assertion, bit is cleared. Set by the primary port master upon completion of primary port initialization.</p> <p>When P_BOOT=1 (primary port retains boot priority), the secondary port master access to PCI Standard BAR configurations at 10h to 1Bh is Retried until the P_PORT_READY bit is set.</p> <p>When P_BOOT=0 and bit is 0, cross-bridge traffic initiated by secondary port is returned with Retry.</p> <p>P_PORT_READY mechanism does not have the above effect if the special fixed-size cross-bridge communication window is enabled by setting the XB_MEM input to 1.</p>	Yes	Yes; Serial EEPROM	—

## 6.2.4.14 Chip, Diagnostic, and Arbiter Control Registers

Register 6-231. (CCNTL; PCI:D8h) Chip Control

Bit	Description	Read	Write	Value after Reset
0	<b>Non-Transparent Configuration Semaphore Mechanism Status.</b> Software can check the Non-Transparent Configuration semaphore mechanism status by way of this bit without taking ownership.	Yes	No	0
1	<b>Memory Write Disconnect Control.</b> Controls when PCI 6540, as a target, Disconnects Memory transactions. Values: 0 = Disconnects on queue full or on a 4-KB boundary 1 = Disconnects on a cache line boundary, when the queue fills or on a 4-KB boundary	Yes	Yes	0
2	<b>Cross-Bridge Memory Window Enable (Non-Transparent Mode).</b> When bit is 1, PCI 6540 automatically claims 16 MB of Memory space. This allows boot-up of the Low-Priority Boot port to proceed without waiting for the Priority Boot port to program the corresponding Memory BARs. If bit is 1, the P_PORT_READY or S_PORT_READY mechanism is not relevant and access to BARs is not Retried. <b>Note:</b> Although the default claims 16 MB, the BARs can be changed by serial EEPROM or software to change the window size.	Yes	Yes	XB_MEM
3	<b>Reserved.</b>	Yes	No	0
4	<b>Secondary Bus Prefetch Disable.</b> Controls PCI 6540 ability to prefetch during downstream Memory Read transactions. Values: 0 = Prefetches and does not forward Byte Enables during Memory Read transactions. 1 = Requests only 1 Dword from the target during Memory Read transactions and forwards Byte Enables. PCI 6540 returns a Target Disconnect to the requesting master on the first Data transfer. Memory Read Line and Memory Read Multiple transactions remain prefetchable.	Yes	Yes	0
5	<b>Reserved.</b>	Yes	No	0
6	<b>Transparent Access.</b> Enables access to Shadow registers (which are also Transparent Mode registers) 44h to 5Fh when operating in Non-Transparent mode.	Yes	Yes	0
7	<b>Reserved.</b>	Yes	No	0

Register 6-232. (DCNTRL; PCI:D9h) Diagnostic Control

Bit	Description	Read	Write	Value after Reset
0	<b>Chip Reset.</b> Chip and secondary bus reset. Setting bit activates full chip reset, asserts S_RSTOUT#, and forces the Bridge Control register Secondary Reset bit to be set (BCNTRL[6]=1; PCI:42h Shadow register). After resetting the PCI 6540 registers, this bit is cleared; however, BCNTRL[6] remains set to 1. Writing 0 has no effect.	Yes	Yes	0
2:1	<b>Test Mode.</b>	Yes	Yes	00b
3	<b>Secondary Reset Output Mask.</b> Values: 0 = P_RSTIN# assertion causes S_RSTOUT# assertion. 1 = P_RSTIN# assertion does not cause S_RSTOUT# assertion. P_RSTIN# assertion does not reset the primary port control logic state machines. Power <b>not</b> Good (PWRGD=0) clears bit to 0.	Yes	Yes	0
4	<b>Primary Reset Output Mask.</b> Values: 0 = S_RSTIN# assertion causes P_RSTOUT# assertion 1 = S_RSTIN# assertion does not cause P_RSTOUT# assertion Power <b>not</b> Good (PWRGD=0) clears bit to 0.	Yes	Yes	—
5	<b>Primary Reset.</b> Forces P_RSTOUT# assertion on primary interface. Values: 0 = Does not force P_RSTOUT# assertion 1 = Forces assertion of 0 at P_RSTOUT# pin <b>Note:</b> The Secondary Reset bit is in Bridge Control register (BCNTRL[6]; PCI:42h Shadow register).	Yes	Yes	0
7:6	<b>Reserved.</b>	Yes	No	00b

Register 6-233. (ACNTRL; PCI:DAh) Arbiter Control

Bit	Description	Read	Write	Value after Reset
7:0	<b>Arbiter Control.</b> Each bit controls whether a secondary bus master is assigned to the high- or low-priority group. Bits correspond to request inputs S_REQ[7:0]#, respectively. Value of 1h assigns the bus master to the high-priority group.	Yes	Yes	0h
8	<b>Reserved.</b>	Yes	Yes	0
9	<b>PCI 6540 Priority.</b> Defines whether PCI 6540 secondary port is in the high- or low-priority group. Values: 0 = Low-priority group 1 = High-priority group	Yes	Yes	1
15:10	<b>Reserved.</b>	Yes	No	0h

### 6.2.4.15 Power Management Capability Registers

**Register 6-234. (PMCAPID; PCI:DCh) Power Management Capability ID**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Power Management Capability ID.</b> PCI-SIG-issued Capability ID for Power Management is 01h.	Yes	No	01h

**Register 6-235. (PMNEXT; PCI:DDh) Power Management Next Capability Pointer**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Next_Cap Pointer.</b> Provides an offset into PCI Configuration space for the Hot Swap capability location in the New Capabilities Linked List.	Yes	No	E4h

Register 6-236. (PMC; PCI:DEh) Power Management Capabilities

Bit	Description	Read	Write	Value after Reset
2:0	<b>Version.</b> Set to 001b, indicating that this function complies with <i>PCI Power Mgmt. r1.1</i> .	Yes	Only if HSSRRC[7]=1; Serial EEPROM	001b
3	<b>PME Clock.</b> Because PCI 6540 does not require the PCI clock for PME#, set this bit to 0.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0
4	<b>Auxiliary Power Source.</b> Because PCI 6540 does not support PME# while in a D <sub>3cold</sub> state, this bit is always set to 0.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0
5	<b>Device-Specific Initialization (DSI).</b> Returns 0, indicating PCI 6540 does not require special initialization.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0
8:6	<b>Reserved.</b>	Yes	No	000b
9	<b>D<sub>1</sub> Support.</b> Returns 1, indicating that PCI 6540 supports the D <sub>1</sub> device power state	Yes	Only if HSSRRC[7]=1; Serial EEPROM	1
10	<b>D<sub>2</sub> Support.</b> Returns 1, indicating that PCI 6540 supports the D <sub>2</sub> device power state	Yes	Only if HSSRRC[7]=1; Serial EEPROM	1
15:11	<b>PME Support.</b> Values: XXXX1b = PME# asserted from D <sub>0</sub> XXX1Xb = PME# asserted from D <sub>1</sub> XX1XXb = PME# asserted from D <sub>2</sub> X1XXXb = PME# asserted from D <sub>3hot</sub> 1XXXXb = PME# asserted from D <sub>3cold</sub>	Yes	Only if HSSRRC[7]=1; Serial EEPROM	01111b



**Register 6-237. (PMCSR; PCI:E0h) Power Management Control/Status**

Bit	Description	Read	Write	Value after Reset
1:0	<b>Power State.</b> Used to determine the current power state of a function and to set the function into a new power state. Values: 00b = D <sub>0</sub> 01b = D <sub>1</sub> 10b = D <sub>2</sub> 11b = D <sub>3hot</sub>	Yes	Yes; Serial EEPROM	00b
7:2	<b>Reserved.</b>	Yes	No	0h
8	<b>PME Enable.</b> Enables the PME# output pin. Values: 0 = PME# output disabled 1 = PME# output enabled  <b>Note:</b> In Transparent mode, P_PME# and S_PME# should be pulled high and not used. PME# output is always P_PME#. In Non-Transparent mode, the PME# output is either P_PME# or S_PME#, depending on the P_BOOT value.	Yes	Yes; Serial EEPROM	0
12:9	<b>Data Select.</b> Returns 0h, indicating PCI 6540 does not return dynamic data.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0h
14:13	<b>Data Scale.</b> Returns 00b when read. PCI 6540 does not return dynamic data.	Yes	No	00b
15	<b>PME Status.</b> Set to 0, because PCI 6540 does not support PME# signaling.	Yes	Yes; Serial EEPROM	0

**Register 6-238. (PMCSR\_BSE; PCI:E2h) PMCSR Bridge Supports Extensions**

Bit	Description	Read	Write	Value after Reset
5:0	<b>Reserved.</b>	Yes	No	0h
6	<b>B<sub>2</sub>/B<sub>3</sub> Support for D<sub>3hot</sub>.</b> Reflects the BPCC_EN input pin state. Value of 1 indicates that when the PCI 6540 is programmed to D <sub>3hot</sub> state, the secondary bus clock is stopped.	Yes	No	—
7	<b>Bus Power Control Enable.</b> Reflects the BPCC_EN input pin state. Value of 1 indicates that the secondary bus Power Management state follows that of the PCI 6540, with one exception—D <sub>3hot</sub> .	Yes	No	—

**Register 6-239. (PMCDATA; PCI:E3h) Power Management Data**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Power Management Data.</b> Serial EEPROM or ROR Write controlled loadable, but Read-Only during normal operation.	Yes	Only if HSSRRC[7]=1; Serial EEPROM	0h

## 6.2.4.16 Hot Swap Capability Registers

**Register 6-240. (HS\_CNTL; PCI:E4h) Hot Swap Control**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Hot Swap Capability ID.</b> PCI-SIG-issued Capability ID for Hot Swap is 06h.	Yes	No	06h

**Register 6-241. (HS\_NEXT; PCI:E5h) Hot Swap Next Capability Pointer**

Bit	Description	Read	Write	Value after Reset
7:0	<b>Next_Cap Pointer.</b> Provides an offset into PCI Configuration space for the VPD capability location in the New Capabilities Linked List.	Yes	No	E8h

**Register 6-242. (HS\_CSR; PCI:E6h) Hot Swap Control/Status**

Bit	Description	Read	Write	Value after Reset
0	<b>Device Hiding Arm (DHA).</b> DHA is set to 1 by hardware when the Hot Swap port PCI RSTIN# becomes inactive and the handle switch remains unlocked. Handle locking clears this bit. Values: 0 = Disarm Device Hiding 1 = Arm Device Hiding	Yes	Yes	0
1	<b>ENUM# Mask Status (EIM).</b> Enables or disables ENUM# assertion. Values: 0 = Enables ENUM# assertion 1 = Masks ENUM# assertion	Yes	Yes	0
2	<b>Pending INSert or EXtract (PIE).</b> Set when INS or EXT is 1 or INS is armed (write 1 to EXT bit). Values: 0 = Neither is pending 1 = Insertion or extraction is in progress	Yes	No	—
3	<b>LED Status (LOO).</b> Indicates whether LED is ON or OFF. Values: 0 = LED is OFF 1 = LED is ON	Yes	Yes	0
5:4	<b>Programming Interface (PI).</b> Hardcoded at 01b—INS, EST, LOO, EIM, PIE, and Device Hiding supported.	Yes	No	01b
6	<b>Extraction State (EXT).</b> Set by hardware, when the ejector handle is unlocked and INS=0.	Yes	Yes/Clr	—
7	<b>Insertion State (INS).</b> Set by hardware when the Hot Swap port RSTIN# is de-asserted, serial EEPROM autoloading is completed, and ejector handle is locked. Writing 1 to EXT bit also arms INS.	Yes	Yes/Clr	—
15:8	<b>Reserved.</b>	Yes	No	0h

## 6.2.4.17 VPD Capability Registers

Register 6-243. (PVPDID; PCI:E8h) Vital Product Data Capability ID

Bit	Description	Read	Write	Value after Reset
7:0	<b>Vital Product Data Capability ID.</b> PCI-SIG-issued Capability ID for VPD is 03h.	Yes	No	03h

Register 6-244. (PVPD\_NEXT; PCI:E9h) Vital Product Data Next Capability Pointer

Bit	Description	Read	Write	Value after Reset
7:0	<b>Next_Cap Pointer.</b> Provides an offset into PCI Configuration space for the PCI-X capability location in the New Capabilities Linked List.	Yes	No	F0h

Register 6-245. (PVPDAD; PCI:EAh) Vital Product Data Address

Bit	Description	Read	Write	Value after Reset
1:0	<b>Reserved.</b>	Yes	No	00b
7:2	<b>VPD Address.</b> Offset into the serial EEPROM to location where data is written and read. PCI 6540 accesses the serial EEPROM at address PVPDAD[7:2]+40h. The 40h offset ensures that VPD accesses do not overwrite the PCI 6540 serial EEPROM Configuration data stored in serial EEPROM locations 00h to 3Fh.	Yes	Yes	0h
14:8	<b>Reserved.</b>	Yes	No	0h
15	<b>VPD Operation.</b> Writing 0 to this bit generates a Read cycle from the serial EEPROM at the VPD address specified in PVPDAD[7:2]. Bit remains at logic 0 until the serial EEPROM cycle is finished, then set to 1. Data for reads is available in the VPD Data register (PVPDATA; PCI:ECh). Writing 1 to this bit generates a Write cycle to the serial EEPROM at the VPD address specified in PVPDAD[7:2]. This bit remains at logic 1, until the serial EEPROM cycle is finished, then cleared to 0. Place Write data into the VPD Data register.	Yes	Yes	0

Register 6-246. (PVPDATA; PCI:ECh) VPD Data

Bit	Description	Read	Write	Value after Reset
31:0	<b>VPD Data (Serial EEPROM Data).</b> The least significant byte of this register corresponds to the byte of VPD at the address specified by the VPD Address register (PVPDAD[7:2]; PCI:EAh). Data is read from or written to PVPDATA, using standard Configuration accesses.	Yes	Yes	0h

## 6.2.4.18 PCI-X Capability Registers

Register 6-247. (PCIXCAPID; PCI:F0h) PCI-X Capability ID

Bit	Description	Read	Write	Value after Reset
7:0	<b>PCI-X Capability ID.</b> PCI-SIG-issued Capability ID for PCI-X is 07h.	Yes	No	07h

Register 6-248. (PCIX\_NEXT; PCI:F1h) PCI-X Next Capability Pointer

Bit	Description	Read	Write	Value after Reset
7:0	<b>Next_Cap Pointer.</b> Provides an offset into PCI Configuration space for the location of the next capability in the New Capabilities Linked List. Set to ACh in Non-Transparent mode to point to the Message Signaled Interrupt Capabilities.	Yes	No	ACh

Register 6-249. (PCIX\_CMD; PCI:F2h) PCI-X Command

Bit	Description	Read	Write	Value after Reset
0	<b>Date Parity Error Recovery Enable.</b> Values: 0 = Asserts P_SERR#, if PCI 6540 P_SERR# detection is enabled, when the Master Data Parity Error bit is set (PCISR[8]=1; Primary PCI:06h, Secondary PCI:46h) 1 = Disables Data Parity error recovery	Yes	Yes	0
1	<b>Enable Relaxed Ordering.</b> Relaxed Ordering attribute <i>cannot</i> be set.	Yes	No	0
3:2	<b>Maximum Memory Read Byte Count.</b> Standard PCI-X control bits. As an application bridge, PCI 6540 does <i>not</i> use the values set here: 00b = 512 bytes 01b = 1 KB 10b = 2 KB 11b = 4 KB	Yes	Yes	00b
6:4	<b>Maximum Outstanding Split Transactions.</b> Standard PCI-X control bits. As an application bridge, PCI 6540 does <i>not</i> use the values set here: 000b = 1 001b = 2 010b = 3 011b = 4 100b = 8 101b = 12 110b = 16 111b = 32	Yes	Yes	011b
15:7	<b>Reserved.</b>	Yes	No	0h

Register 6-250. (PCIXBSR; PCI:F4h) PCI-X Bridge Status

Bit	Description	Read	Write	Value after Reset
2:0	<b>Function Number.</b> Indicates Function Number (the number in AD[10:8] of the address of a Type 0 Configuration Transaction to which this bridge responds). The function uses this number as part of its Requester and Completer IDs (set to 000b). PCI 6540 uses the Bus, Device, and Function Numbers bits to create the Completer ID when responding with a Split Completion to a read of an internal bridge register.	Yes	No	000b
7:3	<b>Device Number.</b> Indicates Device Number (the number in AD[15:11] of the address of a Type 0 Configuration transaction) assigned to PCI 6540.	Yes	No	11111b
15:8	<b>Bus Number.</b> Additional addresses from which the Primary Bus Number register contents (PCIPBNO; PCI:18h; in the Type 01h Configuration Space header) are read.	Yes	No	—
16	<b>64-Bit Device.</b> Indicates the bridge AD bus data width. This bit is the inverse of the DEV64# input. Values: 0 = 32-bit bus data width 1 = 64-bit bus data width	Yes	No	Inverse of DEV64# Input
17	<b>133 MHz Capability.</b> Indicates the bridge primary interface is capable of 133 MHz operations in PCI-X mode. Values: 0 = Device maximum frequency is 66 MHz 1 = Device maximum frequency is 133 MHz	Yes	No	1
18	<b>Split Completion Discarded.</b> Set if PCI 6540 discards a Split Completion, because the requester on the primary bus would not accept it. Values: 0 = Split Completion not discarded 1 = Split Completion discarded	Yes	Yes/Clr	0
19	<b>Unexpected Split Completion.</b> Set if an Unexpected Split Completion with a Requester ID equal to PCI 6540 Primary Bus, Device, and Function Numbers is received on the bridge primary bus. Values: 0 = No Unexpected Split Completion received 1 = Unexpected Split Completion received	Yes	Yes/Clr	0
20	<b>Device Complexity.</b> PCI 6540 is an application bridge.	Yes	No	1
22:21	<b>Designed Maximum Memory Read Byte Count.</b> 1024 bytes.	Yes	No	01b
25:23	<b>Designed Maximum Outstanding Split Transactions.</b> Four outstanding.	Yes	No	011b
28:26	<b>Designed Maximum Cumulative Read Size.</b> Eight ADQs and 1 KB.	Yes	No	000b
29	<b>Received Split Completion Error Message.</b> Set if PCI 6540 receives a Split Completion Message with the Split Completion Error attribute bit set (AD30=0).	Yes	Yes/Clr	0
31:30	<b>Reserved.</b>	Yes	No	00b

**Register 6-251. (PCIXUPSTR; PCI:F8h) PCI-X Upstream Split Transaction**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Split Transaction Capacity.</b> PCI 6540 stores Split Completions for Memory Reads in the same buffer as Split Completions for I/O and Configuration Reads and Writes. Indicates the buffer size, in ADQ numbers, for storing Split Completions for Memory Reads for requesters on the secondary bus addressing completers on the primary bus.	Yes	No	32d
31:16	<b>Split Transaction Commitment Limit or Outstanding ADQ Limit.</b> Indicates the cumulative Sequence size for PCI-X Memory Read transactions forwarded by the PCI 6540 from requesters on the secondary bus addressing completers on the primary bus. Also indicates upstream Split Transaction size of those types the PCI 6540 is allowed to commit to at one time.	Yes	Yes	32d

**Note:** PCIXUPSTR controls bridge buffer behavior for forwarding Split Transactions from a secondary requester to a primary bus completer.

**Register 6-252. (PCIXDNSTR; PCI:FCh) PCI-X Downstream Split Transaction**

Bit	Description	Read	Write	Value after Reset
15:0	<b>Split Transaction Capacity.</b> PCI 6540 stores Split Completions for Memory Reads in the same buffer as Split Completions for I/O and Configuration Reads and Writes. Indicates the buffer size, in ADQ numbers, for storing Split Completions for Memory Reads for requesters on the primary bus addressing completers on the secondary bus.	Yes	No	32d
31:16	<b>Split Transaction Commitment Limit or Outstanding ADQ Limit.</b> Indicates the cumulative Sequence size for PCI-X Memory Read transactions forwarded by PCI 6540 from requesters on the secondary bus addressing completers on the primary bus. Also indicates downstream Split Transaction size of those types the PCI 6540 is allowed to commit to at one time.	Yes	Yes	32d

**Note:** PCIXDNSTR controls the bridge buffer behavior for forwarding Split Transactions from a primary bus requester to a secondary bus completer.

## 7 SERIAL EEPROM

This section describes information specific to the PCI 6540 serial EEPROM interface and use—access, Autoload mode, Universal Non-Transparent mode groups, and data structure.

### 7.1 OVERVIEW

**Important Note:** *Erroneous serial EEPROM data can cause the PCI 6540 to lock the system. Provide an optional switch or jumper to disable the serial EEPROM in board designs.*

The PCI 6540 provides a two-wire interface to a serial EEPROM device. The interface can control an ISSI IS24C02 or compatible part, which is organized as 256 x 8 bits. The serial EEPROM is used to initialize the internal PCI 6540 registers, and alleviates the need for user software to configure the PCI 6540. If a programmed serial EEPROM is connected, the PCI 6540 automatically loads data from the serial EEPROM after P\_RSTIN# de-assertion.

The data structure is defined in Section 7.5.1. The serial EEPROM interface is organized on a 16-bit base in Little Endian format, and the PCI 6540 supplies a 7-bit serial EEPROM Word address.

The following pins are used for the serial EEPROM interface:

- **EEPCLK**—Serial EEPROM clock output
- **EEPDATA**—Serial EEPROM bi-directional serial data pin

**Note:** *The PCI 6540 does not control the serial EEPROM A0h to A2h address inputs. Set the serial EEPROM address inputs to 0.*

### 7.2 SERIAL EEPROM ACCESS

The PCI 6540 can access the serial EEPROM on a Word basis, using the hardware sequencer. Users access one Word data by way of the PCI 6540 Serial EEPROM Control register:

- Serial EEPROM Address (EEPADDR; PCI:55h)
- Serial EEPROM Start/Read/Write Control (EEPCNTRL; PCI:54h)
- Serial EEPROM Data (EEPDATA; PCI:56h)

**Note:** *In Non-Transparent mode, serial EEPROM access is not supported from the secondary bus.*

Before each access, software should check the Auto Mode Cycle in Progress status (EEPCNTRL[0]; PCI:54h, same bit as Start) before issuing the next Start. The following is the general procedure for Read/Write Serial EEPROM accesses:

1. Program the Serial EEPROM Address register (EEPADDR; PCI:55h) with the Word address.
2. **For Writes**—Program Word data to the Serial EEPROM Data register (EEPDATA; PCI:56h).  
**For Reads**—Proceed to the next step.
3. **For Writes**—Set the Serial EEPROM Command and Start bits (EEPCNTRL[1:0]=11b; PCI:54h) to start the Serial EEPROM Sequencer.  
**For Reads**—Set the Start bit (EEPCNTRL[1:0]=01b; PCI:54h) to start the Serial EEPROM Sequencer.
4. When the serial EEPROM read/write is complete, Serial EEPROM Control register (EEPCNTRL[0]=0; PCI:54h).  
**For Writes**—Serial EEPROM Control register bit [0]=0 indicates that the data was successfully written to the serial EEPROM.  
**For Reads**—Serial EEPROM Control register bit [0]=0 indicates that the data was loaded into Serial EEPROM Data register (EEPDATA; PCI:56h) by the Serial EEPROM Sequencer.

### 7.3 SERIAL EEPROM AUTOLOAD MODE

Upon PWRGD or P\_RSTIN# going high, whichever occurs last, the PCI 6540 autoloading the serial EEPROM data into the internal PCI 6540 registers.

The PCI 6540 initially reads the first offset in the serial EEPROM, which should contain a valid signature 1516h. If the signature is correct, register autoload commences. During autoload, the PCI 6540 reads sequential words from the serial EEPROM and writes to the appropriate registers. If a blank serial EEPROM is connected, the PCI 6540 stops loading the serial EEPROM contents after reading the first word, as the serial EEPROM's signature is not valid. Likewise, if no serial EEPROM is connected, the PCI 6540 also stops loading the serial EEPROM contents after attempting to read the first word.

## 7.4 UNIVERSAL NON-TRANSPARENT MODE GROUPS

Serial EEPROM data in Group 4 is loaded when the Serial EEPROM register (offset 02h) bits [4:1]=0111b. Serial EEPROM data in Group 5 can be autoloading only when bits [4:1]=1111b and one of the following conditions is met:

- PCI 6540 is in Non-Transparent mode, or
- PCI 6540 is in Transparent mode with pin PRV\_DEV=1

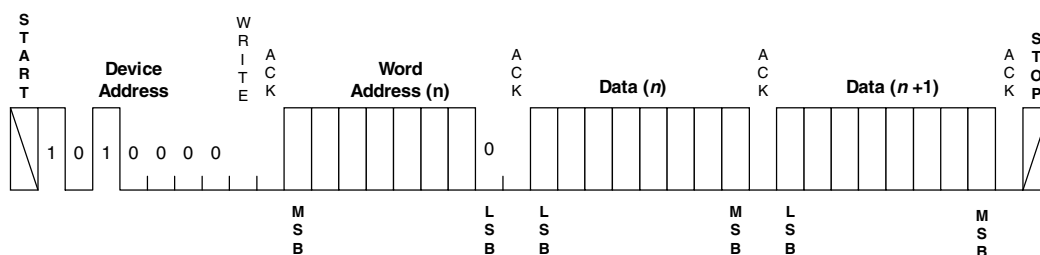
**Caution:** If Group 5 data is not needed in Transparent mode with PRV\_DEV=1, program Group 5 locations in the serial EEPROM to default values.

## 7.5 SERIAL EEPROM DATA STRUCTURE

Following reset and the previously described conditions, the PCI 6540 autoloading the registers with serial EEPROM data. Figure 7-1 describes the serial EEPROM data structure.

The PCI 6540 accesses the serial EEPROM, one word at a time. It is important to note that in the Data phase, bit orders are the reverse of that in the Address phase. The PCI 6540 supports only Serial EEPROM Device Address 0.

**Write:**



**Read:**

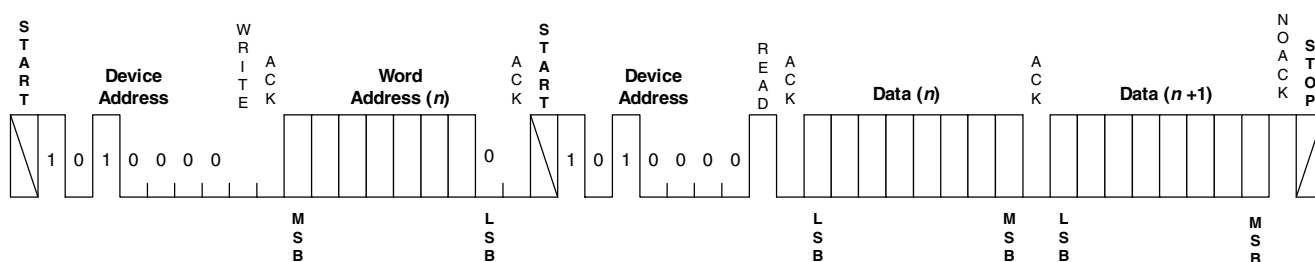


Figure 7-1. Serial EEPROM Data Structure



## 7.5.1 Serial EEPROM Address and Corresponding PCI 6540 Register

Table 7-1. Serial EEPROM Address

Serial EEPROM Byte Address	PCI Configuration Offset	Description
00h – 01h	—	<b>Serial EEPROM Signature.</b> Autoload proceeds only if it reads a value of 1516h on the first word loaded. Value: 1516h = Valid signature; otherwise, disables autoloading.
02h	—	<b>Region Enable.</b> Enables or disables certain regions of the PCI Configuration space from being loaded from the serial EEPROM. Valid combinations are: Bit 0 = <b>Reserved.</b> Bits [4:1] = 0000b = Stops autoload at serial EEPROM offset 03h = Group 1. 0001b = Stops autoload at serial EEPROM offset 13h = Group 2. 0011b = Stops autoload at serial EEPROM offset 23h = Group 3. 0111b = Stops autoload at serial EEPROM offset 27h = Group 4. 1111b = Autoloads all serial EEPROM loadable registers = Group 5. Other combinations are undefined. Bits [7:5] = <b>Reserved.</b>
03h	—	<b>Enable Miscellaneous Functions.</b> Bits [7:0] = <b>Reserved.</b>
<b>End of Group 1</b>		
04h – 05h	00h – 01h	<b>Vendor ID (PCIIDR[15:0]).</b>
06h – 07h	02h – 03h	<b>Transparent Device ID.</b> Non-Transparent Device ID = Transparent ID with inverted bit 0.
08h		<b>Reserved.</b>
09h	09h	<b>Transparent Mode Class Code.</b> Contains low byte of Class Code register (PCICCR).
0Ah – 0Bh	0Ah – 0Bh	<b>Transparent Mode Class Code Higher Bytes.</b> Contains upper bytes of Class Code register.
0Ch	0Eh	<b>Transparent Header Type.</b>
0Dh	09h	<b>Non-Transparent Mode Class Code.</b> Contains low byte of Class Code register.
0Eh – 0Fh	0Ah – 0Bh	<b>Non-Transparent Mode Class Code Higher Bytes.</b> Contains upper bytes of Class Code register.
10h	0Eh	<b>Non-Transparent Header Type.</b>
11h	0Fh	<b>Built-In Self Test (BIST) (PCIBISTR).</b> <i>Not supported. Set to 0.</i>
12h – 13h	50h	<b>Internal Arbiter Control (IACNTRL).</b>
<b>End of Group 2</b>		
14h	44h	<b>Primary Flow-Through Control (PFTCR).</b>
15h	45h	<b>Timeout Control (TOCNTRL).</b>
16h – 17h	46h – 47h	<b>Miscellaneous Options (MSCOPT).</b>
18h	48h	<b>Primary Initial Prefetch Count (PITLPCNT).</b>
19h	49h	<b>Secondary Initial Prefetch Count (SITLPCNT).</b>
1Ah	4Ah	<b>Primary Incremental Prefetch Count (PINPCNT).</b>
1Bh	4Bh	<b>Secondary Incremental Prefetch Count (SINPCNT).</b>
1Ch	4Ch	<b>Primary Maximum Prefetch Count (PMAXCNT).</b>
1Dh	4Dh	<b>Secondary Maximum Prefetch Count (SMAXCNT).</b>
1Eh	4Eh	<b>Secondary Flow-Through Control (SFTCR).</b>
1Fh	E3h	<b>Power Management Data (PMCDATA).</b>

Table 7-1. Serial EEPROM Address (Continued)

Serial EEPROM Byte Address	PCI Configuration Offset	Description
20h – 21h	E0h	<b>Power Management Control/Status (PMCSR).</b>
22h – 23h	DEh	<b>Power Management Capabilities (PMC).</b>
<b>End of Group 3</b>		
24h – 25h	2Ch	<b>Subsystem Vendor ID.</b> PCISVID; Primary PCI:2Ch, Secondary PCI:6Ch (Non-Transparent mode).
26h – 27h	2Eh	<b>Subsystem ID.</b> PCISID; Primary PCI:2Eh, Secondary PCI:6Eh (Non-Transparent mode).
<b>End of Group 4</b>		
28h	—	<b>Reserved.</b>
29h	—	Bits [2:0] = Upstream Address Translation Enable bits (UPSTNE[26:24]; Ext:0Bh). Bit 3 = Upstream BAR 0 I/O bit (UPSBAR0MSK[6]=0; Ext:0Bh). Bits [7:4] = Upstream BAR 0 Translation Address, bits [15:12] (UPSTNBAR0[15:12]; Ext:08h).
2Ah – 2Bh	—	Upstream BAR 0 Translation Address, bits [31:16] (UPSTNBAR0[31:16]; Ext:08h).
2Ch	—	Bit 0 = Upstream BAR 0 Prefetchable bit (UPSBAR0MSK[7]; Ext:0Bh). Bit 1 = Upstream BAR 1 64-bit (UPSBAR1MSK[14]=1; EXT:0Bh). Bit 2 = Upstream BAR 2 Prefetchable bit (UPSBAR2MSK[23]; Ext:0Bh). Bit 3 = Upstream BAR 1 Prefetchable bit (UPSBAR1MSK[15]; Ext:0Bh). Bits [7:4] = Upstream BAR 1 Translation Address, bits [23:20] (UPSTNBAR0[23:20]; Ext:08h).
2Dh	—	Upstream BAR 1 Translation Address, bits [31:24] (UPSTNBAR1[31:24]; Ext:09h).
2Eh – 2Fh	—	Upstream BAR 2 Translation Address, bits [15:0] (UPSTNBAR2[15:0]; Ext:0Ah).
30h – 31h	—	Upstream BAR 2 Translation Address, bits [31:16] (UPSTNBAR2[31:16]; Ext:0Ah).
32h – 33h	—	Bits [4:0] = Upstream BAR 0 Translation Mask (UPSBAR0MSK; Ext:0Bh). Bits [10:5] = Upstream BAR 1 Translation Mask (UPSBAR1MSK; Ext:0Bh). Bits [15:11] = Upstream BAR 2 Translation Mask (UPSBAR2MSK; Ext:0Bh).
34h	—	<b>Reserved.</b>
35h	—	Bits [2:0] = Downstream Address Translation Enable bits (DWNTNE[26:24]; Ext:0Fh). Bit 3 = Downstream BAR 0 I/O bit (DWNBAR0MSK[6]=0; Ext:0Fh). Bits [7:4] = Downstream BAR 0 Translation Address, bits [15:12] (DWNTNBAR0[15:12]; Ext:0Ch).
36h – 37h	—	Downstream BAR 0 Translation Address, bits [31:16] (DWNTNBAR0[31:16]; Ext:0Ch).
38h	—	Bit 0 = Downstream BAR 0 Prefetchable bit (DWNBAR0MSK[7]; Ext:0Fh). Bit 1 = Downstream BAR 1 Prefetchable bit (DWNBAR1MSK[15]; Ext:0Fh). Bit 2 = Downstream BAR 2 Prefetchable bit (DWNBAR2MSK[23]; Ext:0Fh). Bit 3 = Downstream BAR 1 64-bit (DWNBAR1MSK[14]=1; EXT:0Fh). Bit [7:4] = Downstream BAR 1 Translation Address, bits [23:20] (DWNTNBAR1[23:20]; Ext:0Dh).
39h	—	Downstream BAR 1 Translation Address, bits [31:24] (DWNTNBAR1[31:24]; Ext:0Dh).
3Ah – 3Bh	—	Downstream BAR 2 Translation Address, bits [15:0] (DWNTNBAR2[15:0]; Ext:0Eh).
3Ch – 3Dh	—	Downstream BAR 2 Translation Address, bits [31:16] (DWNTNBAR2[31:16]; Ext:0Eh).
3Eh – 3Fh	—	Bits [4:0] = Downstream BAR 0 Translation Mask (DWNBAR0MSK; Ext:0Fh). Bits [10:5] = Downstream BAR 1 Translation Mask (DWNBAR1MSK; Ext:0Fh). Bits [15:11] = Downstream BAR 2 Translation Mask (DWNBAR2MSK; Ext:0Fh).
<b>End of Group 5</b>		

## 8 PCI BUS OPERATION

This section describes PCI transactions to which the PCI 6540 responds and those it initiates when operating with one or both of its interfaces in Conventional PCI mode.

### 8.1 CONVENTIONAL PCI TRANSACTIONS

Table 8-1 lists the Conventional PCI command codes and transaction types to which the PCI 6540 responds and initiates. The *Master* and *Target* columns indicate support for transactions wherein the PCI 6540 initiates transactions as a master, and responds to transactions as a target, on the primary and secondary buses.

Table 8-1. Conventional PCI Transactions

CBE[3:0]#	Transaction Type	Initiates as Master		Responds as Target	
		Primary	Secondary	Primary	Secondary
0000b	Interrupt Acknowledge ( <b>Not Supported</b> )	N	N	N	N
0001b	Special Cycle ( <b>Not Supported</b> )	Y	Y	N	N
0010b	I/O Read	Y	Y	Y	Y
0011b	I/O Write	Y	Y	Y	Y
0100b	<b>Reserved</b>	N	N	N	N
0101b	<b>Reserved</b>	N	N	N	N
0110b	Memory Read	Y	Y	Y	Y
0111b	Memory Write	Y	Y	Y	Y
1000b	<b>Reserved</b>	N	N	N	N
1001b	<b>Reserved</b>	N	N	N	N
1010b	Configuration Read	N	Y	Y	N
1011b	Configuration Write	Type 1	Y	Y	Type 1
1100b	Memory Read Multiple	Y	Y	Y	Y
1101b	Dual Address Cycle (DAC)	Y	Y	Y	Y
1110b	Memory Read Line	Y	Y	Y	Y
1111b	Memory Write and Invalidate	Y	Y	Y	Y

As indicated in Table 8-1, the following Conventional PCI commands are not supported by the PCI 6540. The PCI 6540 reacts to these commands as follows:

- Ignores the commands and does not generate **reserved** command codes.
- 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.
- Does not respond to Special Cycle transactions. To generate Special Cycle transactions on other PCI Buses (downstream or upstream), use a Type 1 Configuration command.
- Does not generate Type 0 Configuration transactions on the primary interface. Responds to Type 0 Configuration transactions on the secondary PCI interface only if Non-Transparent mode is enabled.

## 8.2 SINGLE ADDRESS PHASE

The PCI 6540 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]#.

The PCI 6540 supports only the linear increment Address mode, which is indicated when the lower two Address bits are equal to 00b. If either of the lower two Address bits is equal to a non-zero value, the PCI 6540 automatically Disconnects the transaction after the first Data transfer.

## 8.3 DUAL ADDRESS PHASE

The PCI 6540 supports the Dual Address Cycle (DAC) bus command to transfer 64-bit addresses. In DAC transactions, the first Address phase occurs during the initial FRAME# assertion, and the second Address phase occurs one clock later. During the first Address phase, the DAC command is presented on CBE[3:0]#, and the lower 32 bits of the address on AD[31:0]. The second Address phase retains the cycle command on CBE[3:0]#, and the upper 32 bits of the address on AD[31:0]. When a 64-bit master uses DAC, the master must provide the upper 32 bits of the address on AD[63:32] and the command on CBE[7:4]# during the Address phases of both transactions to allow 64-bit targets additional time to decode the transaction.

DACs are used to access locations that are not in the first 4 GB of PCI Memory space. Addresses in the first 4 GB of Memory space always use a Single Address Cycle (SAC).

The PCI 6540 supports DACs in the downstream and upstream directions. The PCI 6540 responds to DACs for the following commands only:

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

## 8.4 DEVICE SELECT (DEVSEL#) GENERATION

The PCI 6540 performs positive address decoding when accepting transactions on the primary or secondary bus. The PCI 6540 never subtractively decodes. Medium DEVSEL# timing is used for 33 MHz operation and Slow DEVSEL# timing is used for 66 MHz operation.

## 8.5 DATA PHASE

Depending on the command type, the PCI 6540 can support multiple Data phase PCI transactions. Write transactions are treated as Posted Write or Delayed Write transactions.

Table 8-2 lists the forwarding method used for each type of Write operation.

**Table 8-2. Write Transaction Forwarding**

Transaction Type	Forwarding Type
Memory Write	Posted
Memory Write and Invalidate	
I/O Write	Delayed
Type 1 Configuration Write	

### 8.5.1 Posted Write Transactions

When the PCI 6540 determines that a Memory Write transaction is to be forwarded across the bridge, the PCI 6540 asserts DEVSEL# with slow timing and TRDY# in the same cycle, provided that sufficient Buffer space is available in the Posted Write Data queue, and that the queue contains fewer than four outstanding posted transactions. The PCI 6540 can accept one 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 are stored in internal Posted Write buffers and eventually delivered to the target.

The PCI 6540 continues to accept Write data until one of the following occurs:

- Initiator normally terminates the transaction
- Cache line boundary or an aligned 4-KB boundary is reached, depending on transaction type
- Posted Write data buffer fills

When one of the last two events occurs, the PCI 6540 returns a Target Disconnect to the requesting initiator on this Data phase to terminate the transaction.

After the Posted Write transaction is selected for completion, the PCI 6540 requests ownership of the target bus. This can occur while the PCI 6540 is receiving data on the initiator bus. After the PCI 6540 has ownership of the target bus, and the target bus is detected in the idle condition, the PCI 6540 generates the Write cycle and continues to transfer Write data until all Write data corresponding to that transaction is delivered, or a Target Termination is received. If Write data exists in the queue, the PCI 6540 can drive one quad-Dword of Write data each PCI Clock cycle. If Write data is flowing through the PCI 6540 and the initiator stalls, the PCI 6540 inserts wait states on the target bus if the queue empties.

The PCI 6540 ends the transaction on the target bus when one of the following conditions is met:

- All Posted Write data was delivered to the target
- Target returns a Target Disconnect or Retry (the PCI 6540 starts another transaction to deliver the remaining Write data)
- Target returns a Target Abort (the PCI 6540 discards remaining Write data)

The Master Latency Timer expires, and the PCI 6540 no longer retains the target bus grant (the PCI 6540 starts another transaction to deliver the remaining Write data).

### 8.5.2 Memory Write and Invalidate Transactions

Memory Write and Invalidate transactions guarantee the transfer of entire cache lines. By default, the PCI 6540 Retries a Memory Write and Invalidate cycle until there is space for one or more cache lines of data in the internal buffers. The PCI 6540 then completes the transaction on the secondary bus as a Memory Write and Invalidate cycle. The PCI 6540 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, the PCI 6540 Disconnects and completes the Write cycle on the secondary bus as a normal Memory Write cycle by way of the Miscellaneous Options register Memory Write and Invalidate Control bit (MSCOPT[12]; PCI:46h). The PCI 6540 Disconnects Memory Write and Invalidate commands at aligned cache line boundaries. The Cache Line Size register (Transparent mode—PCICLSR; PCI:0Ch, Non-Transparent mode—PCISCLSR; Primary PCI:0Ch, Secondary PCI:4Ch and PCISCLSR; Primary PCI:4Ch, Secondary PCI:0Ch) cache line size value provides the number of Dwords in a cache line. For the PCI 6540 to generate Memory Write and Invalidate transactions, this cache line size value must be written to a value of 08h, 10h, or 20h Dwords. If an invalid cache line size is programmed, wherein the value is 0, not a power of two, or greater than 20h Dwords, the PCI 6540 sets the cache line size to the minimum value of 08h. The PCI 6540 always Disconnects on the Cache Line boundary.

When the Memory Write and Invalidate transaction is Disconnects before a Cache Line boundary is reached (typically because the Posted Write Data buffer fills), the transaction is converted to a Memory Write transaction.

### 8.5.3 Delayed Write Transactions

A Delayed Write transaction forwards I/O Write and Type 1 Configuration cycles by way of the PCI 6540, and is limited to a single quad-Dword Data transfer.

When a Write transaction is first detected on the initiator bus, the PCI 6540 claims the access and returns a Target Retry to the initiator. During the cycle, the PCI 6540 samples the Bus Command, Address, and Address Parity bits. The PCI 6540 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 the PCI 6540 schedules a Delayed Write transaction to be the next cycle to complete based on its ordering constraints, the PCI 6540 initiates the transaction on the target bus. The PCI 6540 transfers the Write data to the target.

If the PCI 6540 receives a Target Retry in response to the Write transaction on the target bus, the PCI 6540 continues to repeat the Write transaction until the Data transfer is complete, or an error condition is encountered. If the PCI 6540 is unable to deliver Write data after  $2^{24}$  attempts (programmable through the Timeout Control register Maximum Retry Counter Control bits, TOCNTRL[2:0]; PCI:45h), the PCI 6540 ceases further write attempts and returns a Target Abort to the initiator. The Delayed transaction is removed from the Delayed Transaction queue.

The PCI 6540 also asserts P\_SERR# if the Command register P\_SERR# Enable bit is set (Transparent mode—PCICR[8]=1; PCI:42h, Non-Transparent mode—PCICR[8]=1; Primary PCI:04h, Secondary PCI:44h). When the initiator repeats the same Write transaction (same command, address, Byte Enable bits, and data), after the PCI 6540 has completed data delivery and retains all complete cycle information in the queue, the PCI 6540 claims the access and returns TRDY# to the initiator, to indicate that the Write data was transferred. If the initiator requests multiple quad-Dwords, the PCI 6540 asserts STOP#, in conjunction with TRDY#, to signal a Target Disconnect. Only those bytes of Write data with valid Byte Enable bits are compared. If any 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 is transferred to the target, the PCI 6540 returns a Target Retry to the initiator. The PCI 6540 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, the PCI 6540 does not make a new entry into the Delayed Transaction queue.

The PCI 6540 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 the primary and secondary Bridge Control register Master Timeout bits (Transparent mode—BCNTRL[8:9]; PCI:3Eh, Non-Transparent mode—BCNTRL[8:9]; PCI:42h Shadow register, respectively), as well as the Timeout Control register Master Timeout Divider bits (Transparent mode only—TOCNTRL[7:4]; PCI:45h). If the Discard Timer expires before the Write cycle is Retried, the PCI 6540 discards the Delayed Write transaction from the Delayed Transaction queue. The PCI 6540 also conditionally asserts P\_SERR#.

### 8.5.4 Write Transaction Address Boundaries

The PCI 6540 imposes internal Address boundaries when accepting Write data. The aligned Address boundaries are used to prevent the PCI 6540 from continuing a transaction over a device Address boundary and to provide an upper limit on maximum latency. When the aligned address boundaries are reached (per conditions listed in Table 8-3), the PCI 6540 returns a Target Disconnect to the initiator.

**Table 8-3. Write Transaction Disconnect Address Boundaries**

Transaction Type	Condition	Aligned Address Boundary
Delayed Write	All	Disconnects after one Data Transfer
Posted Memory Write	Memory Write Disconnect Control Bit = 0 <sup>1</sup>	4-KB Aligned Address Boundary
	Memory Write Disconnect Control Bit = 1 <sup>1</sup>	Disconnects at Cache Line Boundary

1. Memory Write Disconnect Control bit is located in the Chip Control register in Configuration space (Transparent mode—CCNTL[1]; PCI:40h, Non-Transparent mode—CCNTL[1]; PCI:D8h).

### 8.5.5 Buffering Multiple Write Transactions

The PCI 6540 continues to accept Posted Memory Write transactions if space for at least 1 Dword of data in the Posted Write Data buffer remains and there are fewer than four outstanding Posted Memory Write cycles. If the Posted Write Data buffer fills before the initiator terminates the Write transaction, the PCI 6540 returns a Target Disconnect to the initiator.

Delayed Write transactions are posted when one or more open entries exist in the Delayed Transaction queue. The PCI 6540 can queue up to four Posted Write transactions and four Delayed transactions in both the downstream and upstream directions.

### 8.5.6 Read Transactions

Delayed Read forwarding is used for all Read transactions that cross the PCI 6540.

Delayed Read transactions are treated as prefetchable or non-prefetchable.

Table 8-4 delineates the read behavior (prefetchable or non-prefetchable) for each type of Read operation.

**Table 8-4. Read Transaction Prefetching**

Transaction Type	Read Behavior
I/O Read	Never Prefetches
Configuration Read	
Memory Read	Downstream—Prefetches if address is in prefetchable space Upstream—Prefetches if prefetch disable is off (default)
Memory Read Line	Always Prefetches if request is for more than one Data transfer
Memory Read Multiple	

## 8.5.7 Prefetchable Read Transactions

A Prefetchable Read transaction is a Read transaction wherein the PCI 6540 performs speculative DWORD reads, transferring data from the target before the data 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. The PCI 6540 enables all Byte Enable bits of subsequent transfers.

Prefetchable behavior is used for Memory Read Line and Memory Read Multiple transactions, as well as Memory Read transactions that fall into Prefetchable Memory space.

The prefetched data amount depends on the transaction type. The amount of prefetching may also be affected by the amount of free buffer space available in the PCI 6540 and by Read Address boundaries encountered. In addition, there are several PCI 6540-specific registers that can be used to optimize read prefetch behavior.

Prefetching should not be used for those Read transactions that cause side effects on the target device (*that is*, Control and Status registers, FIFOs, and so forth). The target device BARs indicate whether a Memory Address region is prefetchable.

## 8.5.8 Non-Prefetchable Read Transactions

A Non-Prefetchable Read transaction is a Read transaction issued by the initiator into a non-prefetchable region. The transaction is used for I/O and Configuration Read transactions, as well as for Memory Reads from Non-Prefetchable Memory space. In this case, the PCI 6540 requests only 1 Dword from the target and Disconnects the initiator after delivery of the first Dword of Read data.

Use Non-Prefetchable Read transactions for regions where extra Read transactions could have side effects (*such as* in FIFO memory or the Control registers). If it is important to retain the Byte Enable bit values during the Data phase, use Non-Prefetchable Read transactions. If these locations are mapped in Memory space, use the Memory Read command and map the target into Non-Prefetchable (Memory-Mapped I/O) Memory space to utilize non-prefetching behavior.

## 8.5.9 Read Prefetch Address Boundaries

The PCI 6540 imposes internal Read Address boundaries on read prefetching. The PCI 6540 uses the Address boundary to calculate the initial amount of prefetched data. During Read transactions to Prefetchable regions, the PCI 6540 prefetches data until it reaches one of these aligned Address boundaries, unless the target signals a Target Disconnect before reaching the Read Prefetch boundary. After reaching the aligned Address boundary, the PCI 6540 may optionally continue prefetching data, depending on certain conditions. (Refer to Section 18, "PCI Flow-Through Optimization.") When finished transferring Read data to the initiator, the PCI 6540 returns a Target Disconnect with the last Data transfer, unless the initiator completes the transaction before delivering all the prefetched Read data. Remaining prefetched data is discarded.

Prefetchable Read transactions in Flow-Through mode prefetch to the nearest aligned 4 KB Address boundary, or until the initiator de-asserts FRAME#.

Table 8-5 delineates the Read Prefetch Address boundaries for Read transactions during Non-Flow-Through mode.

Table 8-5. Read Prefetch Address Boundaries

Transaction Type	Address Space	Prefetch Aligned Address Boundary
Configuration Read	—	1 Dword (No Prefetch)
I/O Read		
Memory Read	Non-Prefetchable	
Memory Read	Prefetchable	Configured by way of Prefetch Count registers
Memory Read Line		
Memory Read Multiple		



### 8.5.10 Delayed Read Requests

The PCI 6540 treats all Read transactions as Delayed Read transactions (*that is*, the Read request from the initiator is posted into a Delayed Transaction queue). Read data from the target is placed into the Read Data queue directed toward the initiator bus interface and transferred to the initiator when the initiator repeats the Read transaction.

When the PCI 6540 accepts a Delayed Read request, it first samples the Read address, Read Bus command, and address parity. When IRDY# is asserted, the PCI 6540 samples the Byte Enable bits for the first Data phase. This information is entered into the Delayed Transaction queue. The PCI 6540 terminates the transaction by signaling a Target Retry to the initiator. Upon receiving the Target Retry, the initiator must continue to repeat the same Read transaction until at least one Data transfer completes, or until it receives a target response other than a Target Retry (Master or Target Abort).

### 8.5.11 Delayed Read Completion with Target

When a Delayed Read request is scheduled to be executed, the PCI 6540 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 non-prefetchable, the PCI 6540 drives the captured Byte Enable bits during the next cycle. If the transaction is a Prefetchable Read transaction, the PCI 6540 drives the captured (first) Byte Enable bits, followed by 0 for the subsequent Data phases. If the PCI 6540 receives a Target Retry in response to the Read transaction on the target bus, it repeats the Read transaction until at least one Data transfer completes or it encounters an error condition. If the transaction is terminated by way of a normal Master Termination or Target Disconnect after at least one Data transfer is complete, the PCI 6540 does not initiate further attempts to read additional data.

If the PCI 6540 is unable to obtain Read data from the target after  $2^{24}$  attempts (default), the PCI 6540 ceases further read attempts and returns a Target Abort to the initiator. The Delayed transaction is removed from the Delayed Transaction queue. The PCI 6540 also asserts P\_SERR# if the Command

register P\_SERR# Enable bit is set (Transparent mode—PCICR[8]=1; PCI:42h, Non-Transparent mode—PCICR[8]=1; Primary PCI:04h, Secondary PCI:44h).

After receiving DEVSEL# and TRDY# from the target, the PCI 6540 transfers the data stored in the internal Read FIFO, then terminates the transaction. The PCI 6540 can accept 1 Dword/Qword of Read data during each PCI Clock cycle—no master wait states are inserted. The number of Dwords/Qwords transferred during a Delayed Read transaction depends on the conditions delineated in Table 8-5 (assuming no Target Disconnect is received).

### 8.5.12 Delayed Read Completion on Initiator Bus

When the transaction completes on the target bus, the Delayed Read data is at the head of the Read Data queue. When all ordering constraints with Posted Write transactions are satisfied, the PCI 6540 transfers the data to the initiator when the initiator repeats the transaction. For Memory Read transactions, the PCI 6540 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. The PCI 6540 returns a Target Disconnect along with the transfer of the last Dword of Read data to the initiator. If the PCI 6540 initiator terminates the transaction before all Read data is transferred, the remaining Read data in the Data buffers is discarded.

When the master repeats the transaction and starts transferring prefetchable Read data from the Data buffers while the Read transaction on the target bus is 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 from the target to the initiator through the Data buffers, long Read bursts can be sustained. In this case, the Read transaction is allowed to continue until the initiator terminates the transaction, an aligned 4 KB Address boundary is reached, or the buffer fills, whichever occurs first. When the buffer empties, the PCI 6540 reflects the stalled condition to the initiator by de-asserting TRDY# for a maximum of eight clock periods until more Read data is available; otherwise, the PCI 6540 Disconnects the cycle. When the initiator

terminates the transaction, the PCI 6540 de-assertion of FRAME# on the initiator bus is forwarded to the target bus. Any remaining Read data is discarded.

The PCI 6540 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 the Primary Maximum Latency register (PCIPMLR[1:0]; PCI:3Fh). If the initiator does not repeat the Read transaction before the Discard Timer expires, the PCI 6540 discards the Read transaction, discards the Read data from its queues, and conditionally asserts P\_SERR#.

The PCI 6540 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 queued Read transaction, the current Read command is not stored because it is contained in the Delayed Transaction queue.

### 8.5.13 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 only in Configuration space.

In addition to accepting Configuration transactions for initialization of its own Configuration space, the PCI 6540 forwards Configuration transactions for device initialization in hierarchical PCI systems, as well as Special Cycle generation.

During Non-Transparent mode, the PCI 6540 can also accept Configuration transactions on its secondary interface. (Refer to Section 20, "Non-Transparent Mode.")

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. Type 0 Configuration transactions are identified by the Configuration command and the lowest two bits of the address are set to 00b.

Type 1 Configuration transactions are issued when the intended target resides on another PCI Bus, or a Special Cycle is to be generated on another PCI Bus. Type 1 Configuration commands are identified by the Configuration command and the lowest two Address bits are set to 01b.

The Register Number is found in both Type 0 and Type 1 formats and provides 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 multi-function device is to be accessed. For single-function devices, this value is not decoded. Type 1 Configuration transaction addresses also include five bits, designating the Device Number that identifies the target PCI Bus device to be accessed. In addition, the Bus Number in Type 1 transactions specifies the target PCI Bus.

### 8.5.14 PCI 6540 Type 0 Access

The Configuration space is accessed by a Type 0 Configuration transaction on the primary interface. The Configuration space is *not* accessible from the secondary bus. The PCI 6540 responds to a Type 0 Configuration transaction by asserting P\_DEVSEL# when the following conditions are met during the Address phase:

- Bus command is a Configuration Read or Write transaction.
- Lower two Address bits on P\_AD[1:0] must be 01b.
- P\_IDSEL must be asserted.
- PCI 6540 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 Byte Enable bit values.
- Type 0 Configuration Read and Write transactions do not use data buffers (*that is*, these transactions are immediately completed, regardless of the Data buffers state).

The PCI 6540 ignores all Type 0 transactions initiated on the secondary interface.

### 8.5.15 Type 1-to-Type 0 Translation

Type 1 Configuration transactions are specifically used 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.

The PCI 6540 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. The PCI 6540 must convert the Configuration command to a Type 0 format, enabling the secondary bus device to respond to the command. Type 1-to-Type 0 translations are performed only in the downstream direction (*that is*, the PCI 6540 generates a Type 0 transaction only on the secondary bus, and never on the primary bus).

The PCI 6540 responds to a Type 1 Configuration transaction and translates the transaction into a Type 0 transaction on the secondary bus when the following conditions are met during the Address phase:

- Lower two Address bits on P\_AD[1:0] are 01b
- Bus Number in address field P\_AD[23:16] is equal to the Secondary Bus Number register value in Configuration space (PCISBNO; PCI:19h)
- Bus command on P\_CBE[3:0]# is a Configuration Read or Write transaction

When translating a Type 1 transaction to a Type 0 transaction on the secondary interface, the PCI 6540 performs the following translations to the address:

- Sets the lower two Address bits on S\_AD[1:0] to 00b
- Decodes the Device Number and drives the bit pattern specified in Table 8-6 on S\_AD[31:16] for the purpose of asserting the device's IDSEL signal
- Sets S\_AD[15:11] to 0h
- Leaves the Function and Register Number fields unchanged

The PCI 6540 asserts a unique address line, based on the Device Number. These address lines may be used as S\_IDSEL signals. 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 used by the PCI 6540.

The PCI 6540 can assert up to 16 unique address lines to be used as S\_IDSEL signals for up to 16 secondary bus devices, for Device Numbers ranging from 0 to 15. Because of the PCI Bus electrical loading constraints, more than 16 IDSEL signals should not be necessary. However, if more than 15 device numbers are needed, an external method of generating IDSEL lines must be used, and the upper Address bits are *not* asserted. The Configuration transaction is translated and passed from primary-to-secondary bus. If an IDSEL pin is not asserted to a secondary device, the transaction terminates in a Master Abort.

The PCI 6540 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 Configuration cycles are forwarded, address stepping is used, and a valid address is driven on the bus before FRAME# assertion. Type 0 Configuration address stepping is programmable through the Miscellaneous Options register Address Step Control bits (MSCOPT[6:4]; PCI:46h).

**Table 8-6. Device Number to IDSEL S\_AD Pin Mapping**

Device Number	P_AD[15:11]	Secondary IDSEL S_AD[31:16]	S_AD Bit
0	00000b	0000_0000_0000_0001b	16
1	00001b	0000_0000_0000_0010b	17
2	00010b	0000_0000_0000_0100b	18
3	00011b	0000_0000_0000_1000b	19
4	00100b	0000_0000_0001_0000b	20
5	00101b	0000_0000_0010_0000b	21
6	00110b	0000_0000_0100_0000b	22
7	00111b	0000_0000_1000_0000b	23
8	01000b	0000_0001_0000_0000b	24
9	01001b	0000_0010_0000_0000b	25
10	01010b	0000_0100_0000_0000b	26
11	01011b	0000_1000_0000_0000b	27
12	01100b	0001_0000_0000_0000b	28
13	01101b	0010_0000_0000_0000b	29
14	01110b	0100_0000_0000_0000b	30
15	01111b	1000_0000_0000_0000b	31
Special Cycle	1XXXXb	0000_0000_0000_0000b	None

### 8.5.16 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 the PCI 6540 detects a Type 1 Configuration transaction intended for a PCI Bus downstream from the secondary bus, the PCI 6540 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:

- Lower two Address bits on AD[1:0] are equal to 01b
- Bus Number falls in the range defined by the lower limit (exclusive) in the Secondary Bus Number register (PCISBNO; PCI:19h) and upper limit (inclusive) in the Subordinate Bus Number register (PCISUBNO; PCI:1Ah)
- Bus command is a Configuration Read or Write transaction

The PCI 6540 also supports Type 1-to-Type 1 upstream forwarding of Configuration Write transactions to support upstream Special Cycle generation. A Type 1 Configuration command is forwarded upstream when the following conditions are met:

- Lower two Address bits on AD[1:0] are equal to 01b
- Bus Number falls outside the range defined by the lower limit (inclusive) in the Secondary Bus Number register (PCISBNO; PCI:19h) and upper limit (inclusive) in the Subordinate Bus Number register (PCISUBNO; PCI:1Ah)
- Device Number in Address bits AD[15:11] is equal to 1111b
- Function Number in Address bits AD[10:8] is equal to 11b
- Bus command is a Configuration Write transaction
- PCI 6540 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

### 8.5.17 Special Cycles

The Type 1 configuration mechanism is used to generate Special Cycle transactions in hierarchical PCI systems. Special Cycle transactions are ignored by operating 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 downstream or upstream direction.

The PCI 6540 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:

- Lower two Address bits on AD[1:0] are equal to 01b
- Device Number in Address bits AD[15:11] is equal to 1111b
- Function Number in Address bits AD[10:8] is equal to 11b
- Register number in Address bits AD[7:2] is equal to 0h
- Bus Number is equal to the Secondary Bus Number register value in Configuration space (PCISBNO; PCI:19h) for downstream forwarding, or equal to the Primary Bus Number register value in Configuration space (PCIPBNO; PCI:18h) for upstream forwarding
- Bus command on the initiator CBE bus is a Configuration Write command

When the PCI 6540 initiates a 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 Cycle 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 because Special Cycles complete as Master Aborts. After the transaction is completed on the target bus, through Master Abort condition detection, the PCI 6540 responds with TRDY# to the next attempt of the Configuration transaction from the initiator. If more than one Data transfer is requested, the PCI 6540 responds with a Target Disconnect operation during the first Data phase.

## 8.6 TRANSACTION TERMINATION IN CONVENTIONAL PCI MODE

This section describes how the PCI 6540 returns transaction termination conditions to the initiator.

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

- **Normal Termination**—Occurs when the initiator de-asserts FRAME# at the beginning of the last Data phase, and de-asserts IRDY# at the end of the last Data phase in conjunction with TRDY# or STOP# assertion from the target.
- **Master Abort**—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 asserted, the initiator de-asserts FRAME# on the next cycle, then de-asserts IRDY# on the following cycle. IRDY# must be asserted in the same cycle in which FRAME# is de-asserted. If FRAME# was de-asserted, IRDY# can be de-asserted on the next Clock cycle following Master Abort condition detection.

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

- **Normal Termination**—TRDY# and DEVSEL# are asserted in conjunction with FRAME# de-assertion and IRDY# assertion.
- **Target Retry**—STOP# and DEVSEL# are asserted without TRDY# during the first Data phase. No data transfers during the transaction. This transaction must be repeated.
- **Target Disconnect (with Data transfer)**—STOP# and DEVSEL# are asserted with TRDY#. Signals that this is the last Data transfer of the transaction.
- **Target Disconnect (without Data transfer)**—STOP# and DEVSEL# are asserted without TRDY# after previous Data transfers. Indicates that no further Data transfers are made during this transaction.
- **Target Abort**—STOP# is asserted without DEVSEL# and TRDY#. Indicates that the target is never able to complete this transaction. DEVSEL# must be asserted for at least one cycle during the transaction before the Target Abort is signaled.

### 8.6.1 PCI 6540-Initiated Master Termination

As an initiator, the PCI 6540 uses normal termination if DEVSEL# is returned by the target within five Clock cycles of PCI 6540 assertion of FRAME# on the target bus. In this case, the PCI 6540 terminates a transaction when the following conditions are met:

- During Delayed Write transactions, a single Dword/Qword is delivered.
- During Non-Prefetchable Read transactions, a single Dword/Qword is transferred from the target.
- During Prefetchable Read transactions, a prefetch boundary is reached.
- For Posted Write transactions, all Write data for the transaction is transferred from Data buffers to the target.
- For Burst transfers (*except* Memory Write and Invalidate transactions), the Master Latency Timer expires and the PCI 6540 bus grant is de-asserted.
- Target terminates the transaction with a Retry, Disconnect, or Target Abort.
- If the PCI 6540 is delivering Posted Write data when it terminates the transaction because the Master Latency Timer expires, the PCI 6540 initiates another transaction to deliver the remaining Write data. The transaction address is updated to reflect the address of the current Dword to be delivered.

If the PCI 6540 is prefetching Read data when it terminates the transaction because the Master Latency Timer expires, the PCI 6540 does not repeat the transaction to obtain additional data.

### 8.6.2 Master Abort Received by PCI 6540

If the initiator initiates a transaction on the target bus and does not detect DEVSEL# returned by the target within five Clock cycles of FRAME# assertion, the PCI 6540 terminates the transaction as specified through the Bridge Control register Master Abort Mode bit (Transparent mode—BCNTRL[5]; PCI:3Eh, Non-Transparent mode—BCNTRL[5]; PCI:42h Shadow register, respectively).

For Delayed Read and Write transactions, the PCI 6540 can assert TRDY# and return FFFF\_FFFFh for reads, or return a Target Abort. SERR# is also optionally asserted.

When a Master Abort is received in response to a Posted Write transaction, the PCI 6540 discards the Posted Write data and makes no further attempts to deliver the data. The PCI 6540 sets the Status register Received Master Abort bit when the Master Abort is received on the primary bus (Transparent mode—PCISR[13]=1; PCI:06h, Non-Transparent mode—PCISR[13]=1; Primary PCI:06h, Secondary PCI:46h), or the Secondary Status register Received Master Abort bit when the Master Abort is received on the secondary interface (Transparent mode—PCISSR[13]=1; PCI:1Eh, Non-Transparent mode—PCISSR[13]=1; Primary PCI:46h, Secondary PCI:06h).

When the Master Abort Mode bit is set and a Master Abort is detected in response to a Posted Write transaction, the PCI 6540 also asserts P\_SERR#, if enabled (Transparent mode—PCICR[8]=1; PCI:42h, Non-Transparent mode—PCICR[8]=1; Primary PCI:04h, Secondary PCI:44h), but not disabled by the

device-specific P\_SERR# disable for Master Aborts that occur during Posted Write transactions. (Refer to Table 8-7.)

### 8.6.3 Target Termination Received by PCI 6540

When the PCI 6540 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 FRAME# de-assertion)
- Target Retry
- Target Disconnect
- Target Abort

The PCI 6540 controls these terminations using various methods, depending on the type of transaction performed.

**Table 8-7. P\_SERR# Assertion Requirements in Response to Master Abort on Posted Write**

Mode	PCI Offset	Description	Bit	Value
Transparent	06h	Received Target Abort	PCISR[13]	1
Non-Transparent	Primary:06h, Secondary:46h			
Transparent	42h	P_SERR# Enable	PCICR[8]	1
Non-Transparent	Primary:04h, Secondary:44h			
Transparent	64h	Master Abort on Posted Write	PSERRED[4]	0
Non-Transparent	96h			

### 8.6.3.1 Delayed Write Target Termination Response

When the PCI 6540 initiates a Delayed Write transaction, the type of Target Termination received from the target can be returned to the initiator. Table 8-8 delineates the response to each type of Target Termination that occurs during a Delayed Write transaction. The PCI 6540 repeats a Delayed Write transaction until the PCI 6540:

- Completes at least one Data transfer
- Receives a Master Abort
- Receives a Target Abort

The PCI 6540 makes  $2^{24}$  write attempts (default), resulting in a response of Target Retry. After the PCI 6540 makes  $2^{24}$  attempts of the same Delayed Write transaction on the target bus, the PCI 6540 asserts P\_SERR# if the Command register P\_SERR# Enable bit is set and the implementation-specific P\_SERR# Disable bit for this condition is not set. (Refer to Table 8-9.) The PCI 6540 stops initiating transactions in response to that Delayed Write transaction and the Delayed Write request is discarded. Upon a subsequent Write transaction attempt by the initiator, the PCI 6540 returns a Target Abort.

Table 8-8. Response to Delayed Write Target Termination

Target Termination	Response				
Normal	Returns Disconnect to initiator with first Data transfer only if multiple Data phases requested.				
Target Retry	Returns Target Retry to initiator. Continue write attempts to target.				
Target Disconnect	Returns Disconnect to initiator with first Data transfer only if multiple Data phases requested.				
Target Abort	Returns Target Abort to initiator. Sets target interface Status register Received Target Abort bit. Sets initiator interface Status register Signaled Target Abort bit.				
	Mode	Initiator (Primary Bus)	Target (Secondary Bus)	Initiator (Secondary Bus)	Target (Primary Bus)
	Transparent	PCISR[11]=1; PCI:06h	PCISSR[12]=1; PCI:1Eh	PCISR[12]=1; PCI:06h	PCISSR[11]=1; PCI:1Eh
	Non-Transparent	PCISR[11]=1; Primary PCI:06h, Secondary PCI:46h	PCISSR[12]=1; Primary PCI:46h, Secondary PCI:06h	PCISR[12]=1; Primary PCI:06h, Secondary PCI:46h	PCISSR[11]=1; Primary PCI:46h, Secondary PCI:06h

Table 8-9. P\_SERR# Assertion Requirements in Response to Delayed Write

Mode	PCI Offset	Description	Bit
Transparent	42h	P_SERR# Enable	PCICR[8]=1
Non-Transparent	Primary:04h, Secondary:44h		
Transparent	64h	Delayed Configuration or I/O Write Non-Delivery	PSERRED[5]=0
Non-Transparent	96h		



### 8.6.3.2 Posted Write Target Termination Response

When the PCI 6540 initiates a Posted Write transaction, the Target Termination cannot be returned to the initiator. Table 8-10 delineates the response to each type of Target Termination that occurs during a Posted Write transaction.

When a Target Retry or Disconnect is returned and Posted Write data associated with that transaction remains in the Write buffers, the PCI 6540 initiates another Write transaction to attempt to deliver the remaining Write data. In the case of a Target Retry, the same address is 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 current Dword address. 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, the PCI 6540 uses the Memory Write command to deliver the remaining Write data because less than a cache line is transferred in the subsequent Write transaction attempt.

After the PCI 6540 makes  $2^{24}$  write attempts and fails to deliver all the Posted Write data associated with that transaction, the PCI 6540 asserts P\_SERR#, if enabled in the Command register, **and** the device-specific P\_SERR# Disable bit for this condition is **not** set. (Refer to Table 8-11.) The Write data is discarded.

**Table 8-10. Response to Posted Write Target Termination**

Target Termination	Response		
Normal	No additional action.		
Target Retry	Repeats Write transaction to target.		
Target Disconnect	Initiates Write transaction to deliver remaining Posted Write data.		
Target Abort	Sets target interface Status register Received Target Abort bit. Asserts P_SERR#, if enabled, and sets the Primary Status register Signaled System Error bit.		
	Mode	Initiator (Primary Bus)	Target (Secondary Bus)
	Transparent	PCICR[8]=1; PCI:42h PCISR[14]=1; PCI:06h	PCISSR[12]=1; PCI:1Eh
	Non-Transparent	PCICR[8]=1; Primary PCI:04h, Secondary PCI:44h PCISR[14]=1; Primary PCI:06h, Secondary PCI:46h	PCISSR[12]=1; Primary PCI:46h, Secondary PCI:06h

**Table 8-11. P\_SERR# Assertion Requirements in Response to Posted Write Parity Error**

Mode	PCI Offset	Description	Bit
Transparent	42h	P_SERR# Enable	PCICR[8]=0
Non-Transparent	Primary:04h, Secondary:44h		
Transparent	64h	Posted Write Parity Error	PSERRED[1]=0
Non-Transparent	96h		

### 8.6.3.3 Delayed Read Target Termination Response

When the PCI 6540 initiates a Delayed Read transaction, the abnormal target responses can be returned to the initiator. Other target responses depend on the amount of data the initiator requests. Table 8-12 delineates the response to each type of Target Termination that occurs during a Delayed Read transaction.

The PCI 6540 repeats a Delayed Read transaction until the PCI 6540:

- Completes at least one Data transfer
- Receives a Master Abort

- Receives a Target Abort
- Produces 2<sup>24</sup> read attempts, resulting in a response of Target Retry

After the PCI 6540 produces 2<sup>24</sup> attempts of the same Delayed Read transaction on the target bus, the PCI 6540 asserts P\_SERR# if the Command register P\_SERR# Enable bit is set and the implementation-specific P\_SERR# Disable bit for this condition is *not* set. (Refer to Table 8-13.) The PCI 6540 stops initiating transactions in response to that Delayed Read transaction, and the Delayed Read request is discarded. Upon a subsequent Read transaction attempt by the initiator, the PCI 6540 returns a Target Abort.

**Table 8-12. Response to Delayed Read Target Termination**

Target Termination	Response				
Normal	If prefetchable, Target Disconnects only if initiator requests more data than read from target. If non-prefetchable, Target Disconnects on first Data phase.				
Target Retry	Re-initiates Read transaction to target.				
Target Disconnect	If initiator requests more data than read from target, returns Target Disconnect to initiator.				
Target Abort	Returns Target Abort to initiator. Sets target interface Status register Received Target Abort bit. Sets initiator interface Status register Signaled Target Abort bit.				
	Mode	Initiator (Primary Bus)	Target (Secondary Bus)	Initiator (Secondary Bus)	Target (Primary Bus)
	Transparent	PCISR[11]=1; PCI:06h	PCISSR[12]=1; PCI:1Eh	PCISR[12]=1; PCI:06h	PCISSR[11]=1; PCI:1Eh
	Non-Transparent	PCISR[11]=1; Primary PCI:06h, Secondary PCI:46h	PCISSR[12]=1; Primary PCI:46h, Secondary PCI:06h	PCISR[12]=1; Primary PCI:06h, Secondary PCI:46h	PCISSR[11]=1; Primary PCI:46h, Secondary PCI:06h

**Table 8-13. P\_SERR# Assertion Requirements in Response to Delayed Read**

Mode	PCI Offset	Description	Bit
Transparent	42h	P_SERR# Enable	PCICR[8]=1
Non-Transparent	Primary:04h, Secondary:44h		
Transparent	64h	Delayed Read-No Data from Target	PSERRED[6]=0
Non-Transparent	96h		

## 8.6.4 PCI 6540-Initiated Target Termination

The PCI 6540 can return a Target Retry, Disconnect, or Abort to an initiator for reasons other than detection of that condition at the target interface.

### 8.6.4.1 Target Retry

When it cannot accept Write data or return Read data as a result of internal conditions, the PCI 6540 returns a Target Retry to the initiator when any of the following conditions are met:

- **Delayed Write Transactions**

- Transaction is in the process of entering the Delayed Transaction queue.
- Transaction has entered the Delayed Transaction queue, but target response has not been received.
- Target response was received, but the Posted Memory Write Ordering rule prevents the cycle from completing.
- Delayed Transaction queue is full; therefore, transaction cannot be queued.
- Transaction with the same address and command was queued.
- Locked sequence is being propagated across the PCI 6540, and the Write transaction is not a Locked transaction.
- Target bus is locked and the Write transaction is a Locked transaction.

- **Delayed Read Transactions**

- Transaction is in the process of entering the Delayed Transaction queue.
- The Read request was queued, but Read data is not yet available.
- Data was read from the target, but the data is not at the head of the Read Data queue, or a Posted Write transaction precedes it.
- Delayed Transaction queue is full, and the transaction cannot be queued.
- Delayed Read request with the same address and bus command was queued.
- Locked sequence is being propagated across the PCI 6540, and the Read transaction is not a Locked transaction.
- Target bus is locked and the Write transaction is a Locked transaction.

- **Posted Write Transactions**

- Posted Write Data buffer does not contain sufficient space for the address and at least two Qwords of Write data.
- Locked sequence is being propagated across the PCI 6540, and the Write transaction is not a Locked transaction.

When a Target Retry is returned to a Delayed transaction initiator, 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.

### 8.6.4.2 Target Disconnect

The PCI 6540 returns a Target Disconnect to an initiator when the PCI 6540:

- Reaches an internal address boundary
- Reaches a 4 KB boundary for a Posted Memory Write cycle
- Cannot accept further Write data
- Contains no further Read data to deliver

### 8.6.4.3 Target Abort

The PCI 6540 returns a Target Abort to an initiator when the PCI 6540:

- Returns a Target Abort from the intended target
- Detects a Master Abort on the target, and the Master Abort Mode bit is set (Transparent mode—BCNTRL[5]=1; PCI:3Eh, Non-Transparent mode—BCNTRL[5]=1; PCI:42h Shadow register, respectively)
- Cannot obtain Delayed Read data from the target or nor deliver Delayed Write data to the target after  $2^{24}$  attempts

When returning a Target Abort to the initiator, the PCI 6540 sets the Status register Signaled Target Abort bit corresponding to the initiator interface. (Refer to Table 8-14.)

**Table 8-14. Response to Target Abort**

Mode	Initiator (Primary Bus)	Initiator (Secondary Bus)
Transparent	PCISR[11]=1; PCI:06h	PCISR[12]=1; PCI:06h
Non-Transparent	PCISR[11]=1; Primary PCI:06h, Secondary PCI:46h	PCISR[12]=1; Primary PCI:06h, Secondary PCI:46h

## 9 PCI-X BUS OPERATION

This section describes PCI-X transactions that the PCI 6540 responds to as a target and those it initiates as a master when operating with one or both of its interfaces in PCI-X mode.

### 9.1 OVERVIEW

Because the PCI 6540 is a PCI-X bridge (capable of operating in PCI-X mode), each of the PCI 6540 interfaces (primary or secondary) is capable of operating in Conventional PCI mode when a Conventional PCI device is installed (PCI-XCAP is connected to ground).

The PCI 6540 supports PCI-X mode in the following way:

- Source bridge for the primary bus informs the PCI 6540 of the primary bus mode, with the PCI-X initialization pattern at the rising edge of P\_RSTIN#
- Senses S\_PCI-XAP state and properly initializes the secondary bus devices
- Supports 64-bit addressing on both interfaces
- Implements a 64-bit AD Bus on either interface
- Completes all DWORD transactions and Burst Memory Read transactions as Split transactions, if the transaction crosses the PCI 6540 and the originating interface is in PCI-X mode
- PCI 6540 does not support system topologies, Special Cycle, and Interrupt Acknowledge cases (in Conventional PCI nor PCI-X mode)

### 9.2 GENERAL BUS RULES

When operating in PCI-X mode, the PCI 6540 adheres to the following PCI-X transactions rules:

- Address phase is the phase in which FRAME# is asserted. In the Address phase, the AD Bus contains the starting address (*except* for Split Completions) and the CBE Bus contains the command.
- Starting address of Configuration Read and Write transactions is aligned to a DWORD boundary. Split Completion transactions use only a partial starting address.
- Attribute phase follows the Address phase(s).
- In the Target Response phase, the CBE Bus is **reserved** and driven high (the clock after the Attribute phase).
- Burst transactions include the Byte Count (number of bytes between the first byte of the transaction and the last byte of the Sequence, inclusive) in the attributes.
- DWORD transactions do not use a Byte Count and **reserve** the Attribute Byte Count field.
- Target Response phase is one or more clocks after the Attribute phase and ends when the target asserts DEVSEL#. If no target asserts DEVSEL#, there are no Data phases (Master Abort).
- Transactions using the I/O Read and Write and Configuration Read and Write commands (one Data phase transaction) are initiated only as 32-bit transactions (REQ64# is de-asserted, as in Conventional PCI mode).
- For Memory Write transactions, the Byte Count is not adjusted for bytes whose Byte Enables are de-asserted within the transaction (Byte Count is the same).

Table 9-1 lists the definitions for typical Conventional PCI (provided for reference) and PCI-X transaction phases.

Table 9-1. Transaction Phase Definitions

Conventional PCI Transaction Phases	Description	PCI-X Transaction Phases	Description
Address	One clock for SAC. Two clocks for DAC.	Address	One clock for SAC. Two clocks for DAC.
Data	Clocks (after the Address phase) in which Data transfer is allowed.	Attribute	One clock. Provides further details about the transaction.
Initiator Termination	Initiator signals the end of the transaction on the last Data phase.	Target Response	One or more clocks when the target claims the transaction by asserting DEVSEL#.
Turn-Around	Idle clock for changing from one signal driver to another.	Data	Clocks in which Data transfer is allowed.
		Initiator Termination	Initiator signals the end of the transaction one clock before the last Data phase.
		Turn-Around	Idle clock for changing from one signal driver to another.

### 9.2.1 Initiator Rules

When operating in PCI-X mode, the PCI 6540 adheres to the following rules when initiating a transaction:

- Begins a transaction by asserting FRAME# within two clocks after GNT# is asserted and the bus is idle (within six clocks if the transaction uses a Configuration command).
- Asserts IRDY# two clocks after the Attribute phase (in PCI-X, Wait states are not allowed for initiators).
- De-asserts FRAME# one clock before the last Data phase (for four or more Data phases) or two clocks after the target asserts TRDY# for single Data phase (or terminates the transaction with another method).
- De-asserts IRDY# one clock after the last Data phase (for four or more Data phases) or two clocks after the target asserts TRDY# for single Data phase (or terminates the transaction with another method).
- Ends the transaction as a Master Abort if no target asserts DEVSEL# on or before the SUB decode time.

- For Write and Split Completion transactions, the PCI 6540 drives the first data value on the AD Bus two clocks after the Attribute phase and advances to the second data value (for Burst transaction with more than one Data phase) two clocks after the target asserts DEVSEL# (in anticipation of the target asserting TRDY#). If the target inserts wait states (by not asserting TRDY# after DEVSEL# is asserted), the PCI 6540 toggles between the first and second data values until the target asserts TRDY# (or terminates the transaction with another method).
- Terminates a transaction when the byte count is satisfied. Disconnects a Burst transaction (before the byte count is satisfied) only on an ADB.
- The PCI 6540 retains 64 clocks of Latency Timer value. It Disconnects the current transaction on the next ADB if the Latency Timer expires and GNT# is de-asserted.

### 9.2.2 Target Rules

When operating in PCI-X mode, the PCI 6540 adheres to the following rules when responding to a transaction as a target:

- Memory address range for the PCI 6540 (and all devices) is no smaller than 128 bytes and aligned to ADBs.

- Claims the transaction by asserting DEVSEL#, using Decode B.
- Does not signal Wait State after the first Data phase. Signals Split Response, Target Abort, or Retry within eight clocks of FRAME# assertion and signals Single Data Phase Disconnect (only on the first Data phase), Data Transfer, or Disconnect at the next ADB within 16 clocks of FRAME# assertion.
- May signal Target Abort on any Data phase regardless of its relationship to an ADB.
- If DEVSEL#, STOP#, and TRDY# are not de-asserted, the PCI 6540 de-asserts these signals one clock after the last Data phase, then floats them one clock after that.

### 9.2.3 Bus Arbitration Rules

The following are the bus arbitration rules:

- PCI 6540 asserts and de-asserts REQ# on any clock (no requirement to de-assert REQ# after a Target Termination).
- PCI 6540 de-asserts REQ# on any clock, independent of whether GNT# is asserted (without transaction initiation after GNT# is asserted).
- Arbiter asserts GNT# on any clock if all the GNT# signals are de-asserted. GNTx# remain asserted for a minimum of five clocks while the bus is idle or until the initiator asserts FRAME# or de-asserts REQ#.
- If only one device asserts REQ#, the Arbiter holds GNT# asserted to that device.
- If the Arbiter de-asserts GNT# to one device, it must wait until the next clock to assert GNT# to another device.
- Fast Back-to-Back transactions are **not** allowed in PCI-X mode.

### 9.2.4 Configuration Transaction Rules

When operating in PCI-X mode:

- Initiators must drive the address for four clocks before asserting FRAME#
- Transaction must include the target Device Number in AD[15:11] of the Address phase
- Type 0 Configuration Write transaction target stores its Device and Bus Numbers in its internal registers

### 9.2.5 Parity Error Rules

- In PCI-X mode, when the PCI 6540 detects a Data Parity error while receiving data (*for example*, the target of a write or Split Completion, or initiator of a read), the PCI 6540 asserts PERR# on the second clock after PAR and PAR64 are driven (one clock later than Conventional PCI mode).
- During Read transactions, as a target, the PCI 6540 drives PAR and PAR64 on clock  $n+1$  for the Read data it drives on clock  $n$  and the Byte Enables driven by the initiator on clock  $n+1$ .
- During Write transactions, as an initiator, the PCI 6540 drives PAR64 and PAR on clock  $n+1$  for the Write data and Byte Enables it drives on clock  $n$ .
- PCI 6540 asserts SERR# when it detects a Parity error during an Attribute phase.

### 9.2.6 Bus Data Width Rules

- In PCI-X mode, the width of each transaction is determined with a handshake protocol (similar to Conventional PCI protocol) on REQ64# and ACK64#
- PCI 6540 implements 64-bit in both its interfaces and is capable of generating a 64-bit Memory address
- Attribute phase is always a single clock long for both 64- and 32-bit initiators
- Only Burst transactions (Memory commands other than Memory Read DWORD) use 64-bit Data transfers

### 9.2.7 Split Transaction Rules

- Transactions that are terminated with Split Response result in one or more Split Completion transactions
- Split Completions contain Read data or a Split Completion Message, but not both
- If returning Read data, the completer must return all data (the full Byte Count) unless an error occurs
- Requester must accept all Split Completion Data phases, and can terminate a Split Completion with Data Transfer or Target Abort
- If the request is a Write transaction, or if the completer encounters an error while executing the request, the completer sends a Split Completion Message to the requester

## 9.3 PCI-X SEQUENCES

One or more transactions of a single logical transfer is called a *Sequence*. If a Sequence is broken into more than one transaction, the bytes of all these transactions are included in the request Byte Count that started the Sequence. Each transaction in the same Sequence carries the same Requester ID and Tag.

The PCI 6540 does not initiate a new Sequence using the same Tag until the previous Sequence using that Tag is complete.

The Sequence has more than one transaction if it is a Burst write and Disconnected by the initiator or target. The initiator must resume the Sequence by initiating another Burst Write transaction using the same command and adjusting the starting address and Byte Count for the data.

- If the Sequence is a Burst write and the target signals a Target Abort or no target responds (Master Abort), the Sequence ends when the transaction terminates.
- If the Sequence is a Burst read that executes as an Immediate Transaction, the Sequence terminates when the transaction terminates. The requester is not required to resume an immediate read Sequence.

When a Sequence crosses the PCI 6540, the number of transactions within the Sequence on each bus is determined by the device behavior on each bus. The PCI 6540 maintains Split Completions, in address order, for the same Sequence.

## 9.4 ADB AND BUFFER SIZE

An Allowable Disconnect Boundary (ADB) is an aligned 128-byte address [the lower seven bits are zeros (0)]. ADBs are the same, regardless of transaction width. To proceed from one ADB to the next, a Burst transaction requires 16 Data phases on a 64-bit bus and 32 Data phases on a 32-bit bus.

After a Burst Data transfer starts and the target accepts more than a single Data phase, the transaction can only be stopped using one of the following methods:

- Target or Initiator Disconnect at an ADB
- Transaction Byte Count is satisfied
- Target Abort

If a burst transaction is Disconnected (by the initiator or the target) on an ADB, the last byte address transferred is 7Fh (modulo 80h).

## 9.5 DEPENDENCIES BETWEEN AD AND CBE#

PCI-X Memory, I/O, and Configuration spaces are addressable at the byte level as Conventional PCI mode. Bytes appear on byte lanes according to their individual byte address and transfer width, as listed in Table 9-2.

For burst transactions, the PCI-X requester uses the full address bus to indicate the starting byte address (including AD[1:0]) and includes the Attribute Byte Count field.

For I/O and Memory DWORD transactions, the PCI-X initiator uses the full address bus to indicate the starting address (including AD[1:0]). The Attribute Byte Count field is **reserved**. The ending address is always the last byte of the Dword.

**Note:** Interrupt Acknowledge and Special Cycle transactions have no address.

The CBE Bus is **reserved** and driven high throughout Burst transactions (*except* Memory Writes). During the Data phases of all DWORD and Memory Write transactions, the CBE Bus is used as Byte Enables.

Table 9-2. Byte Lanes Assignment

Byte Address AD[2:0]	Byte Lane	
	64-Bit Transfer	32-Bit Transfer
000b	0	0
001b	1	1
010b	2	2
011b	3	3
100b	4	0
101b	5	1
110b	6	2
111b	7	3



## 9.6 PCI-X COMMAND ENCODING

PCI-X command codes are listed in Table 9-3. The initiator must not generate **reserved** commands and targets should ignore transactions using **reserved** commands.

For all commands (*except* DAC), the transaction command appears on CBE[3:0]# during the single Address phase. For transaction with DACs, CBE[3:0]#

contains the DAC command in the first Address phase and the Transaction command in the second Address phase.

The Alias to Memory Read and Write Block commands are not generated by the initiators and treated as Memory Read Block and Memory Write Block, respectively, by targets.

For 64-bit transactions, CBE[7:4]# contain the transaction command in both Address phases.

Table 9-3. PCI-X Command Encoding

CBE[3:0]#	PCI-X Command	Length	Byte Enable Usage
0000b	Interrupt Acknowledge ( <b>Not Supported</b> )	Dword	Valid
0001b	Special Cycle ( <b>Not Supported</b> )	Dword	Valid
0010b	I/O Read	Dword	Valid
0011b	I/O Write	Dword	Valid
0100b	<b>Reserved</b>	N/A	N/A
0101b	<b>Reserved</b>	N/A	N/A
0110b	Memory Read DWORD	Dword	Valid
0111b	Memory Write	Burst	Valid
1000b	Alias to Memory Read Block	Burst	<b>Reserved</b> / Driven High
1001b	Alias to Memory Write Block	Burst	<b>Reserved</b> / Driven High
1010b	Configuration Read	Dword	Valid
1011b	Configuration Write	Dword	Valid
1100b	Split Completion	Burst	<b>Reserved</b> / Driven High
1101b	DAC	N/A	N/A
1110b	Memory Read Block	Burst	<b>Reserved</b> / Driven High
1111b	Memory Write Block	Burst	<b>Reserved</b> / Driven High

## 9.7 ATTRIBUTES

The transaction is further defined by additional information included in the Attribute phase (always single clock, regardless of the address or data transfer width). The upper buses (AD[63:32] and CBE[7:4]#) of 64-bit devices are **reserved** and driven high during the Attribute phase. PCI-X transactions have three attribute formats. The Requester Attribute format applies to all transactions, *except* Type 0 Configuration Read and Write transactions and Split Completion transactions. Figure 9-1 illustrates the Requester Attribute bit assignments.

The Requester ID (AD[23:8]) is the combination of the Requester Bus, Device, and Function Numbers.

The Sequence ID (AD[28:8]) is the combination of the Tag and the Requester ID. Table 9-4 lists the Requester Attribute bit definitions.

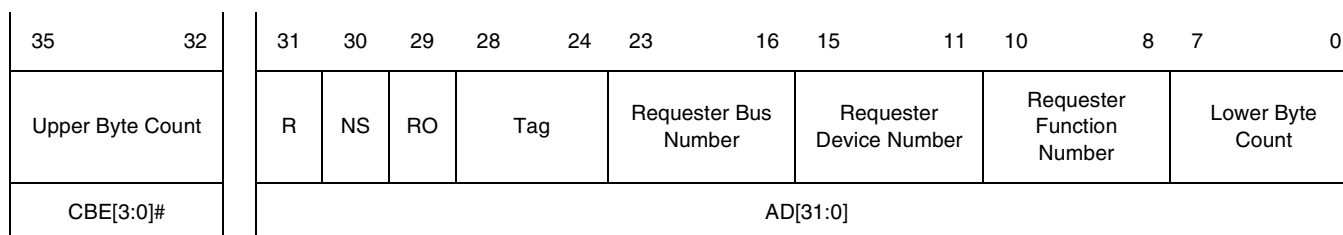


Figure 9-1. Requester Attribute Bit Assignments

Table 9-4. Requester Attribute Bit Definitions

Bit(s)	Attribute	Function										
CBE[3:0]# AD[7:0]	Upper Byte Count Lower Byte Count <i>Example:</i> <table><tr><th>Byte Count[11:0]</th><th>Bytes</th></tr><tr><td>0000_0000_0001b</td><td>1</td></tr><tr><td>0010_0101_0110b</td><td>598</td></tr><tr><td>1111_1111_1111b</td><td>4095</td></tr><tr><td>0000_0000_0000b</td><td>4096</td></tr></table>	Byte Count[11:0]	Bytes	0000_0000_0001b	1	0010_0101_0110b	598	1111_1111_1111b	4095	0000_0000_0000b	4096	Indicates the number of bytes the initiator plans to move in the remainder of the Sequence.  If the transaction is Disconnected for any reason and the initiator continues the Sequence, the initiator must adjust the Byte Count contents in the subsequent transactions of the same Sequence to be the number of bytes remaining in the Sequence.
Byte Count[11:0]	Bytes											
0000_0000_0001b	1											
0010_0101_0110b	598											
1111_1111_1111b	4095											
0000_0000_0000b	4096											
AD[10:8]	Requester Function Number	Contains the requester Function Number (the function in the Configuration address to which the function responds).										
AD[15:11]	Requester Device Number	Contains the Device Number assigned to the requester (from the PCI-X Bridge Status register Device Number bits, PCIXBSR[7:3]; PCI:F4h).										
AD[23:16]	Requester Bus Number	Identifies the requester Bus Number (from the PCI-X Bridge Status register Bus Number bits, PCIXBSR[15:8]; PCI:F4h).										
AD[28:24]	Tag	Uniquely identifies up to 32 Sequences from a single initiator. The initiator assigns a unique Tag to each Sequence that begins before the previous one ends. Tag identifies the appropriate Split Completion transaction.										
AD29	Relaxed Ordering Attribute (RO)	Devices are allowed to set AD29 to load Sequences and must clear it for Control and Status Sequences. Split Read requesters are unaffected by this bit.										
AD30	No Snoop (NS)	If an initiator sets AD30, the initiator guarantees that the locations between the starting and ending address, inclusive, of this Sequence are not stored in a system cache. <ul style="list-style-type: none"><li>If a Write transaction is Disconnected, the initiator must not change this attribute on a subsequent transaction in the same Sequence</li><li>If an immediate Read transaction Disconnects, the Sequence ends</li></ul> PCI 6540 ignores AD30 and forwards it, unmodified, with the transaction.										
AD31	<i>Reserved</i> (R)	Set to 0 by the requester and ignored by the completer.										

## 9.8 BURST TRANSACTIONS

A burst transaction is a transaction that uses one of the following commands:

- **Burst Write transactions**—Memory Write, Memory Write Block, or Alias to Memory Write Block
- **Burst Read transactions**—Memory Read Block or Alias to Memory Read Block commands
- Split Completion

Burst transactions (may be initiated as 64- or 32-bit) transfer data on one or more Data phase, up to that required to satisfy the maximum Byte Count. Burst transactions use the full Address bus (including AD[2:0]) to specify the transaction starting Byte address. Burst transactions can start on one side and end on the other of an ADB, a Memory Page boundary, the first 4 GB Memory Address boundary and so forth.

During Burst transaction Data phases, the CBE Bus is **reserved** and driven high by the initiator for all transactions (*except* Memory Writes).

### 9.8.1 Burst Write and Split Completion

The PCI 6540 executes Burst Write and Split Completion transactions as Immediate transactions, as follows:

- As an initiator, the PCI 6540 is the command source, address, attribute, and data.
- As a target, terminates a Burst Write transaction with Target Abort, Single Data Phase Disconnect, Wait State, Data Transfer, Retry, or Disconnect at the next ADB.
- PCI 6540 terminates a Split Completion transaction with Target Abort, Wait State, or Data Transfer, Retry, or Disconnect at the next ADB. Targets are not allowed to respond with Split Response.
- CBE Bus is **reserved** and driven high after the Address phase of all Burst writes and Split Completion transactions (*except* Memory Writes). All bytes between the starting and ending address, inclusive, are included in these transactions.
- If the target inserts initial wait states, it must do so in pairs of clocks, and the initiator must toggle between the first and second data patterns until the target begins accepting data.

### 9.8.2 Burst Read Transactions

Burst Read transactions use the Memory Read Block or Alias to Memory Read Block commands.

As an initiator, the PCI 6540 is the source of the command, address, and attributes (the completer is the source of the data).

- As a target, the PCI 6540 responds to a Burst Read transaction with Split Response, Target Abort, Single Data Phase Disconnect, Wait State, Data Transfer, Retry, or Disconnect at the next ADB.
- If the PCI 6540 responds (as a target) by signaling Single Data Phase Disconnect, Data Transfer, or Disconnect at the next ADB, Data transfers during the Read transaction.
- If the PCI 6540 responds (as a target) with a Split Response, no Data transfers during the Read transaction but is later transferred in one or more Split Completion transactions.

## 9.9 DWORD TRANSACTIONS

A PCI-X DWORD transaction uses Interrupt Acknowledge, Special Cycle, I/O Read, I/O Write, Configuration Read, Configuration Write, or Memory Read DWORD. The PCI 6540 considers the following PCI-X rules:

- DWORD transactions must be initiated as 32-bit transfers (REQ64# must be de-asserted and AD[63:32], CBE[7:4]#, and PAR64 are not used)
- DWORD transactions retain a single Data phase and affect more than a single Dword
- **Reserved** attributes must be set to 0 by the initiator and ignored by the target
- Byte Enables are not valid until two clocks after the Attribute phase, and PCI-X completer with locations that exhibit read side effects must not start a Read operation until the Byte Enables are valid
- Interrupt Acknowledge has no address, and drives any value on the AD[31:0] bus during the Address phase

## 9.10 DEVICE SELECT TIMING

A PCI-X target claims transactions by asserting DEVSEL#, using one of the timings listed in Table 9-5 (Conventional PCI mode DEVSEL# timing is listed for reference).

**Table 9-5. DEVSEL# Timing**

Decode Speed after Address Phase <sup>1</sup>	Conventional PCI	PCI-X
1 Clock	Fast	N/A
2 Clocks	Medium	Decode A
3 Clocks	Slow	Decode B
4 Clocks	SUB	Decode C
5 Clocks	N/A	N/A
6 Clocks	N/A	SUB <sup>2</sup>

1. Decode speeds are measured from the second Address phase in the case of DAC.

2. If no target asserts DEVSEL# within the SUB decode time, the initiator ends the transaction as a Master Abort.

After asserting DEVSEL#, a target must complete the transaction with one or more Data phases by signaling Split Response, Target Abort, Single Data Phase Disconnect, Wait State, Data transfer, Retry, or Disconnect at the next ADB. The PCI 6540 uses Decode B timing.

## 9.11 WAIT STATES AND TARGET INITIAL LATENCY

When initiating a transaction, the PCI 6540 (and all PCI-X initiators) does not insert wait states. The PCI 6540:

- Asserts IRDY# two clocks after the Attribute phase and drives Write data on the AD Bus or prepares to accept Read data
- After IRDY# is asserted, it continues to assert IRDY# until the transaction ends
- When responding to a transaction, it (and all PCI-X targets) inserts wait states
- Inserts wait states (in pairs of clock for Burst write and Split Completion transactions) only on the initial Data phase
- For transactions of more than one Data phase, it does not signal a wait state

The target initial latency is measured from the clock in which the initiator asserts FRAME#, to the clock in which the target signals something other than a wait state. Table 9-6 delineates the target initial latency for all DEVSEL# timing combinations, number of Address phases, and numbers of wait states.

The maximum number of initial wait states a PCI-X target is allowed to insert depends upon how the target terminates the transaction.

- Target must signal Split Response, Target Abort, or Retry within eight clocks of FRAME# assertion
- Target must signal Single Data Phase Disconnect, Data Transfer, or Disconnect at the next ADB within 16 clocks of FRAME# assertion
- If large numbers of wait states are required, executing the transaction as a Split Transaction provides more efficient bus use
- On a Read transaction, the target is allowed to insert any number of initial wait states up (need not be in pairs for Read transactions) to the maximum number

**Table 9-6. Target Initial Latency**

Wait States	SAC				DAC			
	DEVSEL# Timing				DEVSEL# Timing			
	A	B	C	SUB	A	B	C	SUB
0	3	4	5	7	4	5	6	8
1	4	5	6	8	5	6	7	9
2	5	6	7	9	6	7	8	10
3	6	7	8	10	7	8	9	11
4	7	8	9	11	8	9	10	12
5	8	9	10	12	9	10	11	13
6	9	10	11	13	10	11	12	14
7	10	11	12	14	11	12	13	15
8	11	12	13	15	12	13	14	16
9	12	13	14	16	13	14	15	N/A
10	13	14	15	N/A	14	15	16	N/A
11	14	15	16	N/A	15	16	N/A	N/A
12	15	16	N/A	N/A	16	N/A	N/A	N/A
13	16	N/A	N/A	N/A	N/A	N/A	N/A	N/A

## 9.12 SPLIT TRANSACTIONS

Targets must not assume that the PCI 6540 (as an initiator) repeats a transaction terminated with Retry. Split Transactions improve bus efficiency for transactions accessing targets that exhibit long latency. Split Transactions (in PCI-X mode) replace Delayed Transactions (in Conventional PCI mode). A Split Transaction consists of at least two separate bus transactions—a requester-initiated Split Request, and one or more completer-initiated Split Completions.

The following is the sequence of events that occurs for Non-Memory Write transactions in which the initiator and target reside on opposite sides of the PCI 6540 bridge. These transactions are Memory Read Block, Alias to Memory Read Block, Memory Read DWORD, I/O Read, I/O Write, Configuration Read, or Configuration Write command.

1. Split Request (transaction terminated with a Split Response) occurs.
2. After signaling a Split Response, the PCI 6540 forwards the transaction to the completer side.
3. Completer terminates the transaction after Immediate Completion (Abort) or Split Completion cycle.
4. PCI 6540 initiates a Split Completion transaction to send the received Read data or a Split Completion Message to the requester.
5. PCI 6540 becomes the initiator of the Split Completion transaction, and the requester becomes the target (the requester and PCI 6540 switch roles).

Split Completions for the same Sequence must be initiated in address order.

When the PCI 6540 terminates a Read transaction with Split Response, the PCI 6540 transfers the entire requested Byte Count as a Split Completion. The PCI 6540 provides four ADQs of Buffer space for the Split Completion.

### 9.12.1 Split Completion Transaction

A Split Completion transaction is a Burst transaction (includes the Byte Count in the Attribute phase) that uses the Split Completion command.

The following are the characteristics of Split Completion cycles:

- CBE Bus is **reserved** and driven high during all Data phases
- Split Completion transactions can be initiated as 64- or 32-bit transfers
- Target of a Split Completion is the requester that initiated the Split Request
- Completer stores the Requester ID (Bus, Device, and Function Numbers) from the Split Request Attribute phase
- Requester ID becomes part of the Split Completion address driven on the AD Bus during the Split Completion Address phase
- Requester uses the Requester ID to recognize Split Completions that correspond to its Split Request
- Split Completion attributes carry information about the completer rather than the requester
- Completer adjusts the address and Byte Count (to the portion remaining in the Sequence) each time a Split Completion resumes
- Completer can Disconnect a Split Completion transaction only on an ADB
- Completer must maintain the Split Completion data in address order
- If the Split Request is a DWORD transaction, the Lower Address bits in the Split Completion address are set to 0h and the Byte Count bit in the completer attributes is set to 4h
- If the Split Request is a Write transaction, a Split Completion Message is driven on AD[31:0], regardless of the Byte Enables asserted in the Split Request
- If the Split Request is a Read transaction, data is driven on AD[31:0] during the Split Completion Data phase (only byte lanes corresponding to the Byte Enables in the Split Request contain valid data)

### 9.12.2 Immediate Completion by the Completer

When the PCI 6540 forwards a Split Request, it expects that the completer immediately completes the transaction (*for example*, executes the transaction as an Immediate Transaction rather than a Split Transaction) or for the transaction to end with a Master Abort.

If the completer immediately completes the transaction, the PCI 6540 creates a Split Completion transaction (Split Completion address and completer attributes) to return to the requester.

Split Completions are created in the following way:

- PCI 6540 creates the Split Completion address from the original request in the same way as a completer
- For the Completer Attributes, the PCI 6540 creates the Completer ID for the bus in which the Immediate Completion occurred
- If the Immediate Completion occurred on the primary bus, the PCI 6540 supplies the Bus, Device, and Function Numbers from the PCI-X Bridge Status register (PCIXBSR[15:0]; PCI:F4h)
- If the Immediate Completion occurred on the secondary bus, the PCI 6540 supplies the Bus Number from the Secondary Bus Number register (PCISBNO; PCI:19h) and sets the Device and Function Number bits to 0 (PCIXBSR[7:0]=0h; PCI:F4h)

If the Split Request is a Write transaction and the completer immediately completes it, the PCI 6540 also creates a Split Completion Message for the Split Completion Data phase.

### 9.12.3 Split Completion Address

The Split Completion address is driven on the AD Bus during the Address phase of Split Completion transactions. (Refer to Figure 9-2.) The completer copies all information from the Split Request Address and Attribute phases. Table 9-7 lists the Split Completion Address bit definitions.

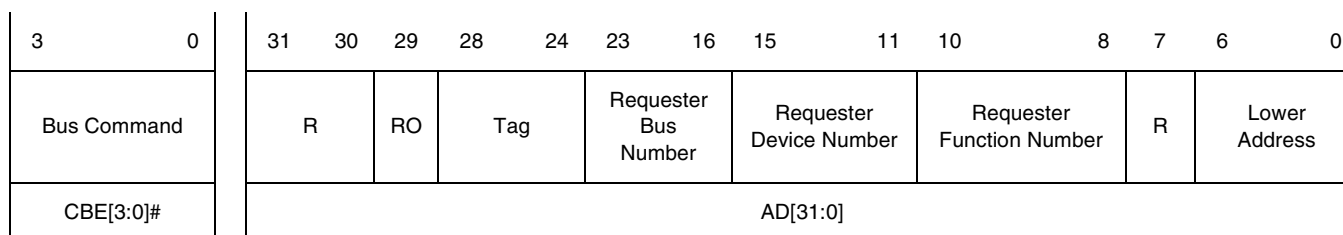


Figure 9-2. Split Completion Address

Table 9-7. Split Completion Address Bit Definitions

Bit(s)	Attribute	Function
AD[6:0]	Lower Address	<p>PCI 6540 uses this information when forwarding a Split Completion to another bus operating in PCI-X mode, to determine where the Split Completion starts relative to an ADB.</p> <p>Completer copies AD[6:0] from the least significant seven bits of the Split Request address, regardless of the command used by the Split Request, if:</p> <ul style="list-style-type: none"> <li>Split Request for this Sequence was a Burst read</li> <li>First Split Completion of the Sequence</li> <li>Split Completion is not a Split Completion Message</li> </ul> <p>AD[6:0] are set to 0h when the Sequence resumes if:</p> <ul style="list-style-type: none"> <li>Split Completion is Disconnected on an ADB</li> <li>Split Request was a DWORD transaction</li> <li>Split Request was a Split Completion Message</li> </ul>
AD7	<b>Reserved (R)</b>	Set to 0 by the target initiator.
AD[10:8]	Requester Function Number	Completer copies AD[10:8] from the corresponding bits of the requester attributes. The requester uses this information to identify the appropriate Split Completions.
AD[15:11]	Requester Device Number	Completer copies AD[15:11] from the corresponding bits of the requester attributes. The requester uses this information to identify the appropriate Split Completions.
AD[23:16]	Requester Bus Number	<p>Completer copies AD[23:16] from the corresponding bits of the requester attributes. The requester uses this information to identify the appropriate Split Completions.</p> <p>PCI 6540 uses this field to identify transactions to forward.</p>
AD[28:24]	Tag	Completer copies AD[28:24] from the corresponding bits of the requester attributes. The requester uses this information to identify the appropriate Split Completions.
AD29	Relaxed Ordering Attribute (RO)	Completer optionally copies AD29 from the corresponding bit of the Requester Attributes Bridges throughout the system, using this bit to influence transaction ordering.
AD[31:30]	<b>Reserved (R)</b>	Set to 00b by the requester and ignored by the completer.
CBE[3:0]#	Bus Command	Contains the Split Completion command.



### 9.12.4 Completer Attributes

The Attribute phase of a Split Completion contains the completer attributes (a combination of the Completer

ID and information about the Sequence stored from the Split Request). (Refer to Figure 9-3 and Table 9-7.)

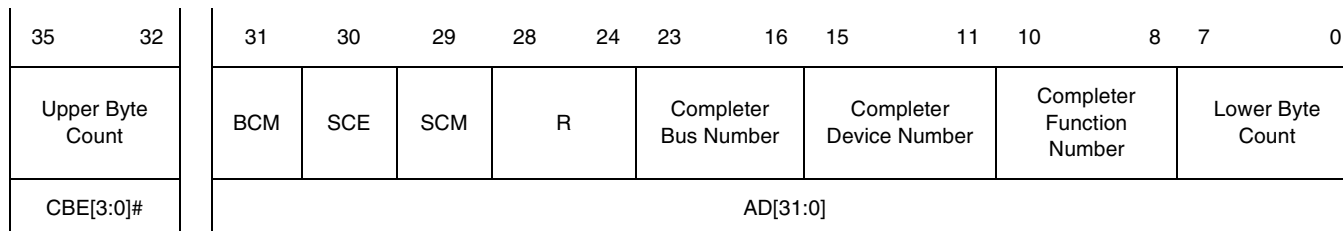


Figure 9-3. Completer Attribute Bit Assignments

Table 9-8. Completer Attribute Bit Definitions

Bit(s)	Attribute	Function										
CBE[3:0]# AD[7:0]	Upper Byte Count Lower Byte Count <i>Example:</i> <table><tr><td><b>Byte Count[11:0]</b></td><td><b>Bytes</b></td></tr><tr><td>0000_0000_0001b</td><td>1</td></tr><tr><td>0010_0101_0110b</td><td>598</td></tr><tr><td>1111_1111_1111b</td><td>4095</td></tr><tr><td>0000_0000_0000b</td><td>4096</td></tr></table>	<b>Byte Count[11:0]</b>	<b>Bytes</b>	0000_0000_0001b	1	0010_0101_0110b	598	1111_1111_1111b	4095	0000_0000_0000b	4096	Indicates the number of bytes the initiator plans to move in this Split Completion. If the transaction is Disconnected for any reason and the initiator continues the Sequence, the initiator must adjust the Byte Count field contents in the subsequent transactions of the same Sequence to be the number of bytes remaining in this Sequence. If the Split Completion is a Split Completion Message, the completer sets the Byte Count to four (0000_0000_0100b).
<b>Byte Count[11:0]</b>	<b>Bytes</b>											
0000_0000_0001b	1											
0010_0101_0110b	598											
1111_1111_1111b	4095											
0000_0000_0000b	4096											
AD[10:8]	Completer Function Number	Contains the Function Number (function in the Configuration address to which the function responds) of the completer within the device.										
AD[15:11]	Completer Device Number	Contains the Device Number assigned to the completer (from the PCI-X Bridge Status register Device Number bits, PCIXBSR[7:3]; PCI:F4h).										
AD[23:16]	Completer Bus Number	Identifies the requester’s Bus Number (from the PCI-X Bridge Status register Bus Number bits, PCIXBSR[15:8]; PCI:F4h).										
AD[28:24]	<b>Reserved</b> (R)	Set to 0h by the requester and ignored by the completer.										
AD29	Split Completion Message (SCM)	Set to 0 by the completer if the Split Completion contains Read data. Set to 1 by the completer if the Split Completion contains a Split Completion Message.										
AD30	Split Completion Error (SCE)	Set to 0 by the completer if the transaction is a Split Completion Message that is an error message.										
AD31	Byte Count Modified (BCM)	Set to 1 by the completer if the Byte Count bit for this Split Completion contains a number smaller than the full remaining transaction Byte Count (to Disconnect the Split Completion at the first ADB). Set to 0 by the completer if the Byte Count bit contains the full remaining Byte Count of the Split Request, or if the Split Completion is a Split Completion Message. <b>Note:</b> The BCM bit is used only for Split Completions resulting from Burst Read transactions (Memory Read Block and Alias to Memory Read Block) and is set to 0 for Split Completions resulting from all other commands.										

### 9.12.5 Accepting Split Completions Requirement

The PCI 6540 accepts all Split Completions resulting from its own Split Requests. The PCI 6540 asserts DEVSEL# on all Split Completions in which the Sequence ID (Requester ID and Tag) corresponds to a PCI 6540-issued Split Request.

If the Split Completion Requester ID is corrupt, it is possible that the ID matches that of an actual system device. The PCI 6540 does not assert DEVSEL# (*for example*, ignores the corrupt transaction) if the Requester ID matches that of the PCI 6540, but the Tag does not match any outstanding requests from the PCI 6540.

When the PCI 6540 asserts DEVSEL# for a Split Completion, it accepts the entire Byte Count requested without signaling Split Response or Single Data Phase Disconnect.

If the PCI 6540 is executing one Split Transaction from one interface (issued Split Response on that interface), the PCI 6540 terminates with Retry Unposted transactions on that interface until the previous Split Transaction completes.

### 9.12.6 Split Completion Messages

Split Completion transactions include a message if the Completer Attributes SCM bit is set (AD29=1). A Split Completion Message is a single DWORD Burst transaction. The Remaining Lower Address bits in the Split Completion address are set to 0 and the Completer Attributes Byte Count is set to 4h for all Split Completion Messages. (Refer to Table 9-9.) The CBE Bus is **reserved** and driven high during the Data phase of a Split Completion Message.

The following illustrates the functions of Split Completion Messages:

- Notifies the requester when a Split Write Request (I/O or Configuration) has completed.
- Indicates error conditions in which delivery of data for a Read request or execution of a Write request is not possible.
- Terminates a Sequence, regardless of the amount of bytes remaining to be sent.
- Split Completion Message Byte Count bit indicates the number of bytes that were not sent for this Sequence (if the Split Request was a Burst read). The remaining Lower Address bits indicate the lower seven bits of the starting address of the remainder of the Sequence. (Refer to Figure 9-4, Table 9-9 and Table 9-10 for further details.)

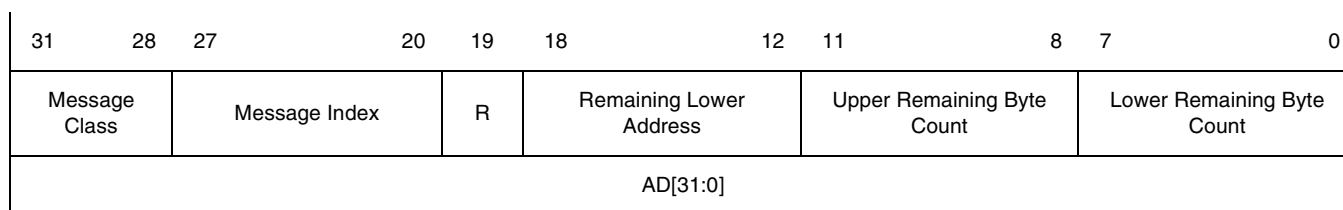


Figure 9-4. Split Completion Message Attribute Bit Assignments

Table 9-9. Split Completion Message Bit Definitions

Bit(s)	Attribute	Function
AD[11:0]	Upper and Lower Remaining Byte Count	If the Split Request was a Burst Memory Read, the completer sets this field to the number of bytes of Read data that were not sent. If the Split Request was a DWORD transaction, the completer sets AD[11:0] to 4h.
AD[18:12]	Remaining Lower Address	If the Split Request was a Burst Memory Read, this field contains the least significant seven bits of the address of the first byte of Read data that was not sent. If the Split Request was a DWORD transaction, the completer sets AD[18:12] to 0h.
AD19	<b>Reserved</b> (R)	Set to 0 by the requester and ignored by the completer.
AD[27:20]	Message Index	Identifies the type of message within the message class. (Refer to Table 9-10 for further details.)
AD[31:28]	Message Class	Split Completion Messages classes. (Refer to Table 9-10 for further details.) Class values: 0000b = Write Completion 0001b = PCI-X Bridge Error 0010b = Completer Error 0011b – 1111b = <b>Reserved</b> <b>Note:</b> The Write Completion class is used for Split Write Completion Messages. The Remaining Lower Address bits are set to 0 (AD[18:12]=0h) and the Upper and Lower Remaining Byte Count bits are set to 4h (AD[11:0]=4h) in the Data phase of this Split Completion Message.

Table 9-10. PCI 6540 Error Message Indices

Index AD[27:20]	Message Class 0 (AD[31:28] = 0000b)	Message Class 1 (AD[31:28] = 0010b)	Message Class 2 (AD[31:28] = 0010b)
00h	Normal Write Completion	<b>Master Abort.</b> PCI 6540 encountered a Master Abort on the destination bus.	Byte Count out of range <sup>1</sup>
01h	<b>Reserved</b>	<b>Target Abort.</b> PCI 6540 encountered a Target Abort on the destination bus.	Split Write Data Parity error <sup>2</sup>
02h	<b>Reserved</b>	<b>Write Data Parity Error.</b> PCI 6540 encountered a Data Parity error on an Unposted Write transaction on the destination bus.	Device-Specific error <sup>3</sup>
03h – FFh	<b>Reserved</b>		

1. The completer uses this message if the sum of the Split Request address and the Byte Count exceed the completer Address range (over device boundary). The completer initiates Split Completion transactions with Read data up to the device boundary.
2. The completer sends this message when a DWORD Write transaction is terminated with Split Response and a Data Parity error detected.
3. The completer uses this message if it encounters an error that prevents execution of the Split Request, and the error is not indicated by one of the other error messages. The PCI 6540 encodes specific error or diagnostic information from the lower four bits of this index.

## 9.13 PCI-X MODE TRANSACTION TERMINATION

The PCI 6540 terminates a transaction using the following methods:

- **Initiator Termination**—Byte Count Disconnection or satisfaction
- **Master Abort Termination**—Target Termination, Disconnection, or Retry; Split Response; or Target Abort Termination

### 9.13.1 PCI 6540 Initiator Termination

#### 9.13.1.1 Byte Count Disconnection or Satisfaction

PCI 6540 Disconnection on an ADB (before the Byte Count is satisfied), and PCI 6540 Termination at the end of the Byte Count, appear the same on the bus.

- PCI 6540 Termination in less than four Data phases occurs only if the starting address, Byte Count, and bus data width are such that the transaction has fewer than four Data phases
- When the PCI 6540 intends to Disconnect a transaction on the first ADB, and the bus data width is such that the starting address is less than four Data phases from the ADB, the PCI 6540 adjusts the Byte Count to terminate on that ADB

### 9.13.1.2 PCI 6540 Master Abort Termination

As an initiator, the PCI 6540 de-asserts FRAME# and IRDY# eight clocks after the Address phase(s) if no target asserts DEVSEL# within six clocks after the Address phase(s).

### 9.13.2 PCI 6540 Target Termination

As a target, after the PCI 6540 asserts DEVSEL# in the Target Response phase, the PCI 6540 completes the transaction with one or more Data phases. On each clock after the target response phase, the PCI 6540 signals its intention with a combination of target control signals (DEVSEL#, STOP#, and TRDY#). A Data phase ends each time the PCI 6540 signals anything other than Wait State. Table 9-11 delineates the alternatives for Data phase signaling.

After the PCI 6540 signals a Data Transfer on one Data phase, the transaction continues until the Byte Count is satisfied or the initiator terminates the transaction. (When the PCI 6540 signals a Split Response or Retry, the transaction immediately terminates, although the Byte Count is not satisfied.)

The transaction immediately terminates if the PCI 6540 signals a Split Response or Retry. Following the transaction termination, the PCI 6540 de-asserts DEVSEL#, STOP#, and TRDY# one clock after the last Data phase (if these signals are not currently de-asserted).

Table 9-11. PCI 6540 Data Phase Signaling

DEVSEL#	STOP#	TRDY#	Data Phase Signaling	Data Transfer	Transaction Status
De-Asserted	De-Asserted	De-Asserted	No Response: Master Abort <sup>1</sup>	No	Terminates
De-Asserted	De-Asserted	Asserted	Split Response	Yes/No <sup>2</sup>	Terminates
De-Asserted	Asserted	De-Asserted	Target Abort	No	Terminates
De-Asserted	Asserted	Asserted	Single Data Phase Disconnect	Yes	Terminates
Asserted	De-Asserted	De-Asserted	Wait State <sup>3</sup>	No	Continues
Asserted	De-Asserted	Asserted	Data Transfer	Yes	Continues
Asserted	Asserted	De-Asserted	Retry	No	Terminates
Asserted	Asserted	Asserted	Disconnect at Next ADB	Yes	Continues to an ADB

1. PCI 6540 does not allow this after asserting DEVSEL#.
2. No Data Transfer on a Split Response for a Read transaction.  
The target latches data on a Split Response for a Write transaction. In both cases, the transaction is not complete until the requester receives the Split Completion.
3. Wait states are allowed only on the first Data phase.

### 9.13.2.1 PCI 6540 Disconnects at Next ADB

The PCI 6540 signals a transaction Disconnection at the next ADB by asserting TRDY#, STOP#, and DEVSEL# on any Data phase of the transaction. After signaling Disconnect at the next ADB, the PCI 6540 can signal only Target Abort on all subsequent Data phases until the transaction ends.

If the PCI 6540 signals Disconnect at the next ADB:

- **Four or more** Data phases before an ADB, the initiator Disconnects the transaction on that ADB
- **After signaling a Data Transfer** on the first Data phase, and the transaction starting address is less than four Data phases from an ADB, the transaction crosses that ADB and continues to the next ADB

### 9.13.2.2 PCI 6540 Retry Termination

The PCI 6540 indicates that it is temporarily unable to complete the transaction by signaling Retry (asserting STOP# and DEVSEL# and holding TRDY# de-asserted on the first Data phase) under the following conditions:

- Initialization time after rising edge of RSTIN# did not elapse
- Buffers for accepting Memory Write or Split Completion transactions are currently full with previous data

The PCI 6540 discards all state information related to a transaction terminated with Retry.

### 9.13.2.3 PCI 6540 Split Response Termination

The PCI 6540 signals a Split Response if it has enqueued the transaction as a Split Request by asserting TRDY#, de-asserting DEVSEL#, and holding STOP# de-asserted on the first Data phase of the transaction. The PCI 6540 drives all AD Bus bits high during the clock in which it signals a Split Response of a Read transaction.

## 9.14 PCI-X MODE BUS AND DATA TRANSFER WIDTH

The PCI 6540 determines the bus data width to which it is attached by the REQ64# state at the rising edge of RSTIN#. PCI-X devices are allowed to implement a 64- or 32-bit interface. Addresses are driven in one or two clocks, depending on whether the transaction uses a memory command and the address is below the first 4-GB boundary. Attributes are always driven in a single clock for 64- and 32-bit devices. As in Conventional PCI mode, the width of a PCI-X Data transfer is negotiated between the initiator and the target on each transaction, using REQ64# and ACK64#.

#### Attributes:

- PCI 6540 is capable of generating addresses greater than 4 GB (when initiating Memory transactions)
- All PCI 6540 Prefetchable Memory Range registers are 64-bit registers
- Split Completions always have a single Address phase for 64- and 32-bit initiators

When executing a transaction with a DAC for a 64-bit transaction, the PCI 6540 drives the entire address (lower address on AD[31:0] and upper address on AD[63:32]) and both commands (DAC on CBE[3:0]# and transaction command on CBE[7:4]#) during the initial Address phase. On the following clock, the PCI 6540 drives the upper address on AD[31:0] (and AD[63:32]) and the transaction command on CBE[3:0]# (and CBE[7:4]#).

The following requirements are applied to all PCI-X devices:

- PCI-X devices support a Status bit, indicating whether they are a 64- or 32-bit device.
- Only Burst transactions use 64-bit transfers. DWORD transactions use 32-bit transfers.
- ADBs are unaffected by the Data transfer width. A 32-bit device has twice as many Data phases between two ADBs.
- AD2 is 0 or 1, depending on the transaction starting Byte address. If AD2 of the starting byte address is 1 (the starting address is in the upper 32 bits of the bus), the 64-bit initiator must drive the data on AD[63:0] and the Byte Enables on CBE[7:0]# of the first Data phase.
- AD[63:32] and CBE[7:4]# are driven high during the Transaction Attribute phase from a 64-bit initiator.
- If the target concurrently asserts ACK64# and DEVSEL# (a 64-bit target), and the target inserts wait states, the initiator must toggle between the first and second QWORD Data phases on AD[63:0] and Byte Enables on CBE[7:0]#. If the transaction starts on an odd Dword, that Dword and its Byte Enables must be copied to the lower half of the bus each time the first Data phase repeats.
- If the target does not assert ACK64# (and the target inserts wait states), the initiator must toggle between the first and second DWORD Data phases of the transaction on AD[31:0] and Byte Enables on CBE[3:0]#.

## 9.15 CONNECTING CONVENTIONAL PCI AND PCI-X INTERFACES

This section provides the PCI 6540 translation of commands and protocol between Conventional PCI and PCI-X interfaces.

### 9.15.1 Conventional PCI Requester, PCI-X Completer

Table 9-12 summarizes the command translation requirements from a Conventional PCI transaction to a PCI-X transaction.

When the PCI 6540 prefetches more than a single Dword, it uses the Memory Read Block command. If a Memory Read Block command is used, the Byte Count is controlled by the PCI 6540 prefetch algorithm. The PCI 6540 Buffer memory writes transactions from its Conventional PCI interface, and counts the number of bytes to be forwarded to the PCI-X interface.

When the PCI 6540 forwards a transaction other than a Memory Write transaction:

- PCI 6540 terminates the transaction on the originating bus with Retry, stored the address, commands, and so forth, and enqueues a delayed request
- After the PCI 6540 finishes the request on the destination bus, it enqueues a delayed completion
- PCI 6540 follows Delayed Transaction rules on the requester side (Conventional PCI Bus) and Split Transaction rules on the completer side (PCI-X Bus)

**Table 9-12. Conventional PCI-to-PCI-X Command Translation**

Conventional PCI Command	PCI-X Command
I/O Read	I/O Read
I/O Write	I/O Write
Configuration Read	Configuration Read
Configuration Write	Configuration Write
Memory Read	Memory Read DWORD or Memory Read Block
Memory Read Line	Memory Read Block
Memory Read Multiple	Memory Read Block
Memory Write	Memory Write or Memory Write Block
Memory Write and Invalidate	Memory Write Block



### 9.15.2 PCI-X Requester, Conventional PCI Completer

Table 9-13 summarizes the command translation requirements from a PCI-X transaction to a Conventional PCI transaction.

The PCI 6540 translates a PCI-X Memory Read Block command into one of three Conventional PCI Memory Read commands, based on the Byte Count and starting address. If the:

- Starting address and Byte Count are such that only one or fewer Dwords are being read, the Conventional PCI transaction uses the Memory Read command
- PCI-X transaction reads more than one Dword, but does not cross a cache line boundary, the Conventional PCI transaction uses the Memory Read Line commands
- PCI-X transaction crosses a cache line boundary, the Conventional PCI transaction uses the Memory Read Multiple command

If PCI-X Memory Write Block command (or Memory Write command) Byte Enables are set and the command starts and ends on a cache line boundary, the PCI 6540 translates the command to the Memory Write command on the Conventional PCI interface.

When forwarding a transaction other than a Memory Write, the PCI 6540:

- Terminates the transaction on the originating bus with Split Response. After executing the transaction on the Conventional PCI interface, the PCI 6540 creates the Split Completion to return to the PCI-X requester.
- Follows Split Transaction rules on the requester side (PCI-X Bus) and Delayed Transaction rules on the completer side (Conventional PCI Bus).

**Table 9-13. PCI-X-to-Conventional PCI Command Translation**

PCI-X Command	Conventional PCI Command
I/O Read	I/O Read
I/O Write	I/O Write
Configuration Read	Configuration Read
Configuration Write	Configuration Write
Memory Read DWORD	Memory Read
Memory Read Block	Memory Read, Memory Read Line, or Memory Read Multiple
Memory Write	Memory Write or Memory Write and Invalidate
Memory Write Block	Memory Write or Memory Write and Invalidate



## 10 ADDRESS DECODING

This section describes address decoding, including address ranges, memory address decoding, ISA mode, VGA and private device support, and address translation.

### 10.1 OVERVIEW

The PCI 6540 uses three Address ranges that control I/O and Memory Transaction forwarding across the bridge. These address ranges are defined by Base and Limit Address registers in Configuration space.

### 10.2 ADDRESS RANGES

The PCI 6540 uses the following address ranges to determine which I/O and Memory transactions are forwarded from the primary-to-secondary PCI Bus, and from the secondary-to-primary bus:

- One 32-Bit I/O Address range
- One 32-Bit Memory-Mapped I/O (non-prefetchable memory) range
- One 64-Bit Prefetchable Memory Address range

Transaction addresses falling within these ranges are forwarded downstream from the primary-to-secondary PCI Bus. Transaction addresses falling outside these ranges are forwarded upstream from the secondary-to-primary PCI Bus.

#### 10.2.1 I/O Address Decoding

The PCI 6540 uses the following mechanisms, defined in Configuration space, to specify the I/O Address space for downstream and upstream forwarding:

- I/O Base and Limit Address registers (Base—PCIIOBAR; PCI:1Ch and PCIIOBARU16; PCI:30h, Limit—PCIIOBMT; PCI:1Dh and PCIIOBMTU16; PCI:32h)
- ISA Enable bit (Transparent mode—BCNTRL[2]; PCI:3Eh, Non-Transparent mode—BCNTRL[2]; PCI:42h)
- VGA Enable bit (Transparent mode—BCNTRL[3]; PCI:3Eh, Non-Transparent mode—BCNTRL[3]; PCI:42h)
- VGA Palette Snoop Enable bit (Transparent mode—PCICR[5]; PCI:04h, Non-Transparent mode—PCICR[5]; Primary PCI:04h, Secondary PCI:44h)

To enable I/O transaction downstream forwarding, the Command register I/O Space Enable bit must be set (Transparent mode—PCICR[0]=1; PCI:04h, Non-Transparent mode—PCICR[0]=1; Primary PCI:04h, Secondary PCI:44h). If the I/O Space Enable bit is not set, I/O transactions initiated on the primary bus are ignored. To enable I/O transaction upstream forwarding, the Command register Master Enable bit must be set (Transparent mode—PCICR[2]=1; PCI:04h, Non-Transparent mode—PCICR[2]=1; Primary PCI:04h, Secondary PCI:44h). If the Master Enable bit is not set, the PCI 6540 ignores 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 when there are ongoing I/O transactions on the secondary bus, the PCI 6540 response to the secondary bus I/O transactions is not predictable. Configure the I/O Base and Limit Address registers, and ISA Enable, VGA Enable, and VGA Palette Snoop Enable bits before setting the I/O Space Enable and Master Enable bits, and subsequently change these registers only when the primary and secondary PCI Buses are idle.*

#### 10.2.1.1 I/O Base and Limit Address Registers

The PCI 6540 implements one set of I/O Base and Limit Address registers in Configuration space that define an I/O Address range for downstream forwarding. The PCI 6540 supports 32-bit I/O addressing, which allows I/O addresses downstream of the PCI 6540 to be mapped anywhere in a 4-GB 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-to-secondary PCI Bus. I/O transactions with addresses that fall outside this range are forwarded upstream from the secondary-to-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 (no I/O transactions are forwarded downstream).

The I/O Base register consists of an 8-bit field (PCIIOBAR; PCI:1Ch) and a 16-bit field (PCIIOBARU16; PCI:30h). The upper four bits of the 8-bit field define bits [15:12] of the I/O Base address. The lower four Read-Only bits are hardcoded to 0001b to indicate that the PCI 6540 supports 32-bit I/O addressing. Bits [11:0] of the Base address are assumed to be 0h, which naturally aligns the base address to a 4-KB boundary with a minimum granularity of 4 KB. The 16 bits contained in the I/O Base Upper 16 Bits register (PCIIOBARU16; PCI:30h) define AD[31:16] of the I/O base address. All 16 bits are read/write. After a primary bus or chip reset, the I/O Base address value is initialized to 0000\_0000h.

The I/O Limit register consists of an 8-bit field (PCIIOLMT; PCI:1Dh) and a 16-bit field (PCIIOLMTU16; PCI:32h). The upper four bits of the 8-bit field define bits [15:12] of the I/O limit address. The lower four Read-Only bits are hardcoded to 0001b 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 4-KB I/O Address block. The 16 bits contained in the I/O Limit Upper 16 Bits register (PCIIOLMTU16; PCI:32h) define AD[31:16] of the I/O Limit address. All 16 bits are read/write. After a primary bus or chip reset, the I/O Limit address value is reset to 0000\_0FFFh.

**Note:** Write these registers with their appropriate values before setting the Command register Master or I/O Space Enable bit (Transparent mode—PCICR[2 or 0]=1; PCI:04h, Non-Transparent mode—PCICR[2 or 0]=1; Primary PCI:04h, Secondary PCI:44h, respectively).

### 10.3 MEMORY ADDRESS DECODING

The PCI 6540 has three mechanisms for defining Memory Address ranges for forwarding of Memory transactions:

- Memory-Mapped I/O Base and Limit Address registers (PCIMBAR; PCI:20h and PCIMLMT; PCI:22h, respectively)
- Prefetchable Memory Base and Limit Address registers (Base—PCIPMBAR; PCI:24h and PCIPMBARU32; PCI:28h, Limit—PCIPMLMT; PCI:26h and PCIPMLMTU32; PCI:26h)
- VGA mode (Transparent mode—BCNTRL[3]=1; PCI:3Eh, (Non-Transparent mode—BCNTRL[3]=1; PCI:42h Shadow register)

This subsection describes the first two mechanisms. VGA mode is described in Section 10.5.1.

To enable downstream forwarding of Memory transactions, the Command register Memory Space Enable bit must be set (Transparent mode—PCICR[1]=1; PCI:04h, Non-Transparent mode—PCICR[1]=1; Primary PCI:04h, Secondary PCI:44h). To enable upstream forwarding of Memory transactions, the Command register Master Enable bit must be set (Transparent mode—PCICR[2]=1; PCI:04h, Non-Transparent mode—PCICR[2]=1; Primary PCI:04h, Secondary PCI:44h). Setting the Master Enable bit also enables 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 when there are ongoing memory transactions 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 Enable bit before setting the Memory Space Enable and Master Enable bits, and subsequently change these registers only when the primary and secondary PCI Buses are idle.

### 10.3.1 Memory-Mapped I/O Base and Limit Address Registers

Memory-mapped I/O is also referred to as Non-Prefetchable memory. The Memory-Mapped I/O Base and Limit Address registers define an Address range that the PCI 6540 uses to determine when to forward Memory commands. The PCI 6540 forwards a Memory transaction from the primary-to-secondary interface if the transaction address falls within the Memory-Mapped I/O Address range. The PCI 6540 ignores Memory transactions initiated on the secondary interface that fall into this Address range. Transactions that fall outside this Address range are ignored on the primary interface and forwarded upstream from the secondary interface (provided that the transactions do not fall into the Prefetchable Memory range, or are not forwarded downstream by the VGA mechanism).

The Memory-Mapped I/O Address range supports only 32-bit addressing. *P-to-P Bridge r1.1* 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 1 MB and a maximum range of 4 GB.

The Memory-Mapped I/O Address range is defined by a 16-bit Memory-Mapped I/O Base Address register (PCIMBAR; PCI:20h) and a 16-bit Memory-Mapped I/O Limit Address register (PCIMLMT; PCI:22h). The upper 12 bits of each of these registers correspond to bits [31:20] of the Memory address. The lower four bits are hardcoded to 0h. The lower 20 bits of the Memory-Mapped I/O Base address are assumed to be 0h, which results in a natural alignment to a 1-MB boundary. The lower 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 1-MB block.

**Note:** Write these registers with their appropriate values before setting the Command register Memory Space Enable or Master Enable bit.

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.

#### 10.3.1.1 Prefetchable Memory Base and Limit Address Registers

Locations accessed in the prefetchable memory address range must have true memory-like behavior and not exhibit side effects when read (*that is*, extra reads to a prefetchable memory location must have no side effects). The PCI 6540 prefetches for all types of Memory Read commands in this Address space.

The PCI 6540 Prefetchable Memory Base and Limit Address registers define an Address range that the PCI 6540 uses to determine when to forward Memory transactions. The PCI 6540 forwards a Memory transaction from the primary-to-secondary interface, if the transaction address falls within the Prefetchable Memory Address range. The PCI 6540 ignores memory transactions initiated on the secondary interface that fall into this address range. The PCI 6540 does not respond to transactions that fall outside this address range on the primary interface and forwards those transactions upstream from the secondary interface (provided that the transactions do not fall into the Memory-Mapped I/O Address range, or are not forwarded by the VGA mechanism).

The PCI 6540 Prefetchable Memory range supports 64-bit addressing and provides additional registers to define the upper 32 bits of the Prefetchable Memory Base and Limit addresses. For address comparison, a Single Address Cycle (SAC; 32-bit address) Prefetchable Memory transaction is treated as a 64-bit Address transaction, where the upper 32 bits of the address are equal to 0h. This upper 32-bit value of 0h is compared to the Prefetchable Memory Base and Limit Address Upper 32 Bits registers. The Prefetchable Memory Base Address Upper 32 Bits register must be 0h to pass SAC transactions downstream.

The Prefetchable Memory Address range is defined by a 16-bit Prefetchable Memory Base Address register (PCIPMBAR; PCI:24h) and a 16-bit Prefetchable Memory Limit Address register (PCIPMLMT; PCI:26h). The upper 12 bits of each of these registers correspond to bits [31:20] of the Memory address. The lower four Read-Only bits are hardcoded to 1h, indicating 64-bit address support. The lower 20 bits of the Prefetchable Memory Base address are assumed to be 0h, which results in a natural alignment to a 1-MB boundary. The lower 20 bits of the Prefetchable

Memory Limit address are assumed to be F\_FFFFh, which results in an alignment to the top of a 1-MB block. The maximum Memory Address range is 4 GB for 32-bit addressing, and  $2^{64}$  bytes for 64-bit addressing.

**Note:** Write these registers with their appropriate values before setting the Command register Memory Space Enable or Master Enable bit.

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 (*that is*, 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 to a value greater than that of the lower Limit register; otherwise, the upper 32-bit base must be greater than the upper 32-bit limit.

## 10.4 ISA MODE

The PCI 6540 supports ISA mode by providing an ISA Enable bit in the Bridge Control register in Configuration space (Transparent mode—BCNTRL[2]=1; PCI:3Eh, Non-Transparent mode—BCNTRL[2]=1; PCI:42h Shadow register). ISA mode modifies the PCI 6540 response inside the I/O Address range to support mapping of I/O space in the presence of an ISA Bus in the system. This bit only affects the PCI 6540 response when the following conditions are met:

- Transaction falls inside the Address range defined by the I/O Base and Limit Address registers, and
- Address also falls inside the first 64 KB of I/O space (Address bits [31:16] are 0h)

When the ISA Enable bit is set, the PCI 6540 does **not** forward downstream I/O transactions addressing the upper 768 bytes of each aligned 1 KB block. Only those transactions addressing the lower 256 bytes of an aligned 1 KB block inside the Base and Limit I/O Address range are forwarded downstream. Transactions above the 64-KB I/O Address boundary are forwarded, as defined by the Address range in the I/O Base and Limit registers.

Additionally, if the ISA Enable bit is set, the PCI 6540 forwards upstream those I/O transactions addressing the upper 768 bytes of each aligned 1 KB block within the first 64 KB of I/O space. The Command Configuration register Master Enable bit must also be set (Transparent mode—PCICR[2]=1; PCI:04h, Non-Transparent mode—PCICR[2]=1; Primary PCI:04h, Secondary PCI:44h) to enable upstream forwarding. All other I/O transactions initiated on the secondary bus are forwarded upstream if the transactions fall outside the I/O Address range.

When the ISA Enable bit is set, devices downstream of the PCI 6540 can have I/O space mapped into the first 256 bytes of each 1 KB segment below the 64-KB boundary, or anywhere in I/O space above the 64-KB boundary.

## 10.5 VGA SUPPORT

The PCI 6540 provides two modes for VGA support:

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

### 10.5.1 VGA Mode

When a VGA-compatible device exists downstream from the PCI 6540, enable VGA mode by setting the Bridge Control register VGA Enable bit (Transparent mode—BCNTRL[3]=1; PCI:3Eh, Non-Transparent mode—BCNTRL[3]=1; PCI:42h Shadow register). When operating in VGA mode, the PCI 6540 forwards downstream those transactions that address the VGA Frame Buffer Memory and VGA I/O registers, regardless of the I/O Base and Limit Address register values. The PCI 6540 ignores transactions initiated on the secondary interface addressing these locations.

The VGA Frame buffer resides in the Memory Address range—000A\_0000h to 000B\_FFFFh.

Read transactions to frame buffer memory are treated as non-prefetchable. The PCI 6540 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 I/O addresses 3B0h to 3BBh and 3C0h to 3DFh.

These I/O addresses are aliased every 1 KB throughout the first 64 KB of I/O space [*that is*, Address bits [15:10] are not decoded and can be any value, while Address bits [31:16] must be all zeros (0)].

VGA BIOS addresses starting at C\_0000h are **not** decoded in VGA mode.

### 10.5.2 VGA Snoop Mode

The PCI 6540 provides VGA Snoop mode, allowing for VGA Palette Write transactions to be forwarded downstream. This mode is used when a graphics device downstream from the PCI 6540 needs to snoop or respond to VGA Palette Write transactions. To enable the mode, set the Command register VGA Palette Snoop Enable bit (Transparent mode—PCICR[5]; PCI:04h, Non-Transparent mode—PCICR[5]; Primary PCI:04h, Secondary PCI:44h). The PCI 6540 claims VGA Palette Write transactions by asserting DEVSEL# in VGA Snoop mode.

When the VGA Palette Snoop Enable bit is set, the PCI 6540 forwards downstream transactions with I/O addresses 3C6h, 3C8h, and 3C9h.

These addresses are also forwarded as part of the previously described VGA Compatibility mode. Again, Address bits [15:10] are **not** decoded, while Address bits [31:16] must be equal to 0 (*that is*, these addresses are aliased every 1 KB throughout the first 64 KB of I/O space).

**Note:** *If both the VGA Enable and VGA Palette Snoop Enable bits are set, the PCI 6540 behaves as if only the VGA Enable bit is set.*

## 10.6 PRIVATE DEVICE SUPPORT

In Transparent mode, the PCI 6540 can support PCI devices that are not visible to primary port hosts or masters. These devices are referred to as *Private Devices* and occupy Private Memory space.

By pulling up the PRV\_DEV input pin to 1, the PCI 6540 enables use of Private Memory space at power-up. The Private Memory range must be setup by driver software. The PCI 6540 does not respond to accesses to this Private Memory range on the primary or secondary port.

When PRV\_DEV=1, in conjunction with the Private Memory range, S\_IDSEL using S\_AD[23:16] is also rerouted to S\_AD24. If S\_AD24 is not connected to a PCI device, Type 1 accesses by the primary host to secondary PCI devices using S\_AD[23:16] as IDSEL are Master Aborted on the secondary port.

The Private Device and Memory Enable bits (CCNTL[3:2]; PCI:40h, respectively) are initialized by the PRV\_DEV input pin state. However, software can change their values after reset.

## 10.7 ADDRESS TRANSLATION

The PCI 6540 provides an address translation mechanism to accommodate various memory maps on the primary and secondary buses. Address translation is supported for both downstream and upstream PCI cycles. When enabled, PCI cycles accessing a specific address range on the initiator bus pass through to the target bus as the same type of transaction, but with a different address, as specified by the Address Translation Control registers. (Refer to Section 6.1.2.19, “Address Translation Control Registers,” and Section 6.2.4.13, “Address Translation Control Registers.”)

### 10.7.1 Base Address Registers

The PCI 6540 supports a maximum of three address ranges that can be translated. The resource configuration is performed through three Base Address registers—BAR 0, 1, and 2. Each Base Address register (BAR) has a 32-bit Translation and an 8-bit Configuration register associated with it. BAR 0 can be configured as a 32-bit I/O or Memory BAR. BAR 1 and BAR 2 are 32-bit BARs, but can optionally be configured as a single 64-Bit Memory BAR. Each follows the standard BAR definition described in *PCI r2.3*. There are two sets of these registers—one each for downstream and upstream translation.

Each Base Address register has a programmable Translation Address register. If translation is enabled, the PCI 6540 uses these registers to translate the address of each cycle accessing Memory or I/O space, specified in one of the Base Address registers.

### 10.7.2 Transparent Mode Address Translation

Address translation is not normally used when the PCI 6540 is configured to operate in Transparent mode. If address translation is not required, the registers referred to in this subsection should remain in their default state.

Address translation for transparent PCI-to-PCI bridges is a non-standard function and the method of operation when using this feature is specific to PLX devices.

When operating in Non-Transparent mode, the secondary bus, including the PCI 6540, must be configured by an intelligent host. This may be an issue in embedded systems where address translation is required, but the systems on the secondary bus do not retain the ability to enumerate a PCI system.

When using Transparent mode address translation, all PCI Bus enumeration is controlled by the primary bus host, thereby allowing address translation in systems with non-intelligent devices attached to the secondary bus.

#### 10.7.2.1 Transparent Mode Address Translation Method

The method of address translation used by the PCI 6540 when configured in Transparent mode is similar to Non-Transparent mode.

In this case, the host must be on the bridge primary bus, because the PCI 6540 does not respond to Type 0 Configuration accesses from the secondary bus.

Address translation between the bridge primary and secondary ports is configured using Downstream and Upstream BARs and downstream and upstream Translation Address and Mask BARs. The Base address is programmed into the appropriate BAR and the region size is determined by the appropriate Translation Mask register.

Note that the PCI 6540 only performs address translation if the address on the primary or secondary PCI Bus falls within one of the Address Translation regions. In addition, for primary-to-secondary address translation, the Address Translation regions must be within the standard Memory Base (PCIMBAR;

PCI:20h) and Memory Limit (PCIMLMT; PCI:22h) or the host-configured I/O Base (PCIIOBAR; PCI:1Ch) and I/O Limit (PCIOLMT; PCI:1Dh) regions. For secondary-to-primary address translation, the Address Translation regions must be outside the host-configured Memory and I/O regions.

#### 10.7.2.2 Register Access

As address translation in transparent PCI-to-PCI bridges is a non-standard operation, access to the registers may require additional software to be written.

Tables 10-1 (Transparent and Non-Transparent modes) and 10-2 (Transparent mode only) list the Extended and Shadowed registers required for address translation configuration.

The Extended registers are accessed using the Extended Register Index (EXTRIDX; PCI:D3h) and Extended Register Data (EXTRDATA; PCI:D4h).

The Address Translation BARs occupy locations that are **reserved** or used by other registers in the standard PCI-to-PCI Bridge Configuration header. To allow the BARs to be programmed without overwriting the standard PCI-to-PCI bridge information access to these registers, the PCI 6540 is protected by bits in the Hot Swap Switch and ROR Control register (HSSRRC[6:5]; PCI:9Ch). This also ensures that normal PCI-to-PCI bridge configuration does not accidentally set up address translation.

The Downstream BARs can be accessed using normal PCI Configuration Read/Write cycles if the Hot Swap Switch and ROR Control register Downstream Translation BAR Access bit is set to 1 (HSSRRC[5]=1; PCI:9Ch).

The Upstream BARs can be accessed using normal PCI Configuration Read/Write cycles if the Hot Swap Switch and ROR Control register Upstream Translation BAR Access bit is set to 1 (HSSRRC[6]=1; PCI:9Ch).

To avoid operational issues, reset HSSRRC[6:5] to 00b after the BARs are configured.



Table 10-1. Extended Register Map (Used in Address Translation)—Offset from Extended Register Index

Extended Register Index	3124231615870								Writable	Serial EEPROM Writable
08h	Upstream BAR 0 Translation Address								Yes	Yes
09h	Upstream BAR 1 Translation Address								Yes	Yes
0Ah	Upstream BAR 2 or Upstream BAR 1 Upper 32 Bits Translation Address								Yes	Yes
0Bh	Upstream Translation Enable	Upstream BAR 2 Translation Mask		Upstream BAR 1 Translation Mask		Upstream BAR 0 Translation Mask		Yes	Yes	
0Ch	Downstream BAR 0 Translation Address								Yes	Yes
0Dh	Downstream BAR 1 Translation Address								Yes	Yes
0Eh	Downstream BAR 2 or Downstream BAR 1 Upper 32 Bits Translation Address								Yes	Yes
0Fh	Downstream Translation Enable	Downstream BAR 2 Translation Mask		Downstream BAR 1 Translation Mask		Downstream BAR 0 Translation Mask		Yes	Yes	

**Note:** When the serial EEPROM is set to initialize for Universal Non-Transparent mode applications, these registers also activate translation in Universal Transparent mode if *PRV\_DEV*=1. (Refer to Section 10.7.2, “Transparent Mode Address Translation,” on page 10-6 and Section 10.7.3, “Non-Transparent Mode Address Translation,” on page 10-11 for further details.)

Table 10-2. PCI Configuration Shadowed Registers (Used in Address Translation)—Transparent Mode

PCI Configuration Register Address Primary Offset	To ensure software compatibility with other versions of the PCI 6540 family and to ensure compatibility with future enhancements, write 0 to all unused bits.	PCI Writable	Serial EEPROM Writable
	31 0		
10h	Downstream I/O BAR 0	Only if HSSRRC[5]=1; PCI:9Ch	No
14h	Downstream Memory BAR 1	Only if HSSRRC[5]=1; PCI:9Ch	No
18h	Downstream Memory BAR 2 or Downstream Memory BAR 1 Upper 32 Bits	Only if HSSRRC[5]=1; PCI:9Ch	No
10h	Upstream I/O or Memory BAR 0	Only if HSSRRC[6]=1; PCI:9Ch	No
14h	Upstream Memory BAR 1	Only if HSSRRC[6]=1; PCI:9Ch	No
18h	Upstream Memory BAR 2 or Upstream Memory BAR 1 Upper 32 Bits	Only if HSSRRC[6]=1; PCI:9Ch	No

**Note:** The registers listed in Table 10-2 can be written to only when the appropriate Hot Swap Switch and Read-Only register bit is set to 1 (HSSRRC[6 or 5]=1; PCI:9Ch). To allow normal device operation, HSSRRC[6:5] must be reset to 00b after the registers are configured.

### 10.7.2.3 Address Translation on Primary-to-Secondary (Downstream) Transactions

The standard PCI-to-PCI Bridge Memory and I/O Base and Limit registers specify the Address range within the primary PCI Memory map and I/O Memory map, to which the PCI 6540 responds. Transactions within these regions are forwarded to the secondary bus.

The PCI 6540 retains two Memory regions and one I/O region configured for address translation. These regions must fall within the Base/Limit range of the previously noted Memory and I/O spaces.

To program the Downstream BARs, set the Hot Swap Switch and ROR Control register Downstream Translation BAR Access bit (HSSRRC[5]=1; PCI:9Ch). (Refer to Section 10.7.2.2.)

The Base addresses of the three Downstream Memory regions are configured using the Downstream Memory BARs (Transparent mode—PCIBAR0; Primary PCI:10h, PCIBAR1; Primary PCI:14h and PCIBAR2; Primary PCI:18h, Non-Transparent mode—PCIBAR0; Primary PCI:10h, Secondary PCI:50h, PCIBAR1; Primary PCI:14h, Secondary PCI:54h and PCIBAR2; Primary PCI:18h, Secondary PCI:58h. Program these registers with the lowest (Base) address of the region in which memory translation is to occur. To ensure correct decoding, the Base address must be a multiple of the range (or size) of the Address Translation region. (*For example*, if the range is 1 MB, then suitable Base addresses would be 0000\_0000h, 0010\_0000h, 0020\_0000h, and so forth.)

The Downstream Translation Mask registers (DWNBAR1MSK[15:8], Ext:0Fh and DWNBAR2MSK [23:16], Ext:0Fh) determine the range (or size) of the Address Translation regions. The range must be a power of 2; therefore, the range is specified by programming the appropriate power in the Mask register. (*For example*, 4 MB =  $2^{22}$ , so if a range of 4 MB is required, program the value 22d into the appropriate Mask register.)

When an Address Translation region is mapped into Memory space, the minimum range is 1 MB for DWNBAR1MSK and 4 KB for DWNBAR2MSK. Smaller ranges may be programmed into these

registers; however, the smaller range may result in unpredictable device behavior.

PCIBAR1 and PCIBAR2 may be combined to allow 64-bit address translation. In this case, PCIBAR1 represents the lower 32 bits, and PCIBAR2 the upper 32 bits, of the Base address. To enable 64-bit address translation, set DWNBAR1MSK[14]=1.

DWNBAR0MSK[7] (Non-Transparent mode only), DWNBAR1MSK[15], and DWNBAR2MSK[23] are used to determine whether a region is prefetchable. This allows the Address Translation region to be mapped into Prefetchable Memory space rather than Memory space.

Program DWNTNBAR1; Ext:0Dh and DWNTNBAR2; Ext:0Eh with the Base address for address translation on the secondary bus. Whatever the transaction offset above PCIBARx, the offset is added to DWNTNBARx to obtain the transaction address on the secondary bus. *For example*, if PCIBAR1=1000\_0000h and DWNTNBAR1=8000\_0000h, a primary bus transaction at location 1000\_0100h results in a secondary bus transaction at location 8000\_0100h.

The method of programming the registers for I/O address translation is the same as that for the Memory regions. In this case, the I/O region Base addresses are configured using PCIBAR0; Primary PCI:10h. As with the Memory regions, to ensure correct decoding, program the register with the lowest (Base) address of the region in which I/O translation is to occur, and the Base address as a multiple of the range (or size) of the Address Translation region.

The I/O Address Translation region range is programmed using DWNBAR0MSK[7:0], Ext:0Fh. The minimum range for correct operation is 4 KB. DWNBAR0MSK[7:6] must be set to 00b to ensure that the region is mapped into I/O space, which is not prefetchable.

Program DWNTNBAR0; Ext:0Ch with the Base address for I/O address translation on the secondary bus.

After the appropriate registers are programmed, the individual base address regions may be enabled for address translation by setting the appropriate Downstream Translation Enable bits (DWNTNE[26:24]=111b; Ext:0Fh). DWNTNE[26:24]

enable address translation on PCIBAR2, PCIBAR1, and PCIBAR0, respectively.

#### 10.7.2.4 Address Translation on Secondary-to-Primary (Upstream) Transactions

When performing downstream address translation, the transaction address must be within the range associated with the standard PCI-to-PCI Bridge Memory and I/O Base and Limit registers for address translation to occur. In the case of upstream address translation, the transaction on the secondary bus must be outside of this region. If the transaction is within this region, the cycle is forwarded to the primary bus with the original address (*that is*, no address translation occurs).

The method of operation for upstream address translation is the same as downstream translation. However, there are three memory translation windows as Upstream PCIBAR0 (PCIUBAR0; PCI:10h) can be configured to map into Memory or I/O space. As previously stated, these registers must be configured such that the regions for address translation are outside the PCI-to-PCI Bridge Memory and I/O Base and Limit regions.

To program the Upstream BARs, set the Hot Swap Switch and ROR Control register Upstream Translation BAR Access bit (HSSRRC[6]=1; PCI:9Ch). (Refer to Section 10.7.2.2.)

The Base addresses of the three Upstream Memory regions are configured using PCIUBAR0; PCI:10h, PCIUBAR1; PCI:14h, and PCIUBAR2; PCI:18h. Program these registers with the lowest (Base) address of the region in which memory translation is to occur. To ensure correct decoding, the Base address must be a multiple of the range (or size) of the Address Translation region.

The Upstream BAR 0, 1, and 2 Translation Mask registers (UPSBARxMSK; Ext:0Bh) determine the range (or size) of the Address Translation regions.

To configure PCIUBAR0 to map into Memory space, set UPSBAR0MSK[6]=1.

The minimum range is 1 MB for UPSBAR1MSK and 4 KB for UPSBAR0MSK and UPSBAR2MSK. Smaller ranges may be programmed into these registers;

however, the smaller range may result in unpredictable device behavior

PCIUBAR1 and PCIUBAR2 may be combined to allow 64-bit address translation. In this case, PCIUBAR1 represents the lower 32 bits, and PCIUBAR2 the upper 32 bits, of the Base address. To enable 64-bit address translation, set UPSBAR1MSK[14]=1.

UPSBAR0MSK[7], UPSBAR1MSK[15], and UPSBAR2MSK[23] are used to determine whether a region is prefetchable. This allows the Address Translation region to be mapped into Prefetchable Memory space rather than Memory space.

Program UPSTNBAR0; Ext:08h, UPSTNBAR1; Ext:09h, and UPSTNBAR2; Ext:0Ah with the Base address for address translation on the primary bus. Whatever the transaction offset above PCIUBAR, add the transaction offset to the UPSTNBAR offset to obtain the transaction address on the primary bus.

The method of programming registers for I/O address translation is the same as that for the Memory regions. In this case, the I/O region Base addresses are configured using PCIUBAR0; Primary PCI:10h. As with the Memory regions, to ensure correct decoding, program the register with the lowest (Base) address of the region in which I/O translation is to occur, and the Base address as a multiple of the range (or size) of the Address Translation region.

The range of the I/O Address Translation region is programmed using UPSBAR0MSK[7:0]; Ext:0Bh. The minimum range for correct operation is 4 KB. UPSBAR0MSK[7:6] must be set to 00b to map the region into I/O space and state that the region is not prefetchable.

Program UPSTNBAR0; Ext:08h with the Base address for I/O address translation on the secondary bus.

After the appropriate registers (previously noted) are programmed, the individual base address regions may be enabled for address translation by setting the appropriate bits of UPSTNE[26:24]; Ext:0Bh. Bits [26:24] enable address translation on PCIUBAR2, PCIUBAR1, and PCIUBAR0, respectively. Setting bit [26, 25, or 24] to 1 enables address translation on that BAR.

### 10.7.2.5 Serial EEPROM Configuration of Transparent Mode Address Translation

In Transparent mode, the PCI 6540 loads the serial EEPROM contents up to the end of Group 4 (refer to Section 7, “Serial EEPROM”). The Address Translation Control registers contained in Group 5 of the serial EEPROM must be configured from the primary bus host.

However, if PRV\_DEV is tied high, all serial EEPROM contents, including those in Group 5, are loaded at boot-up and, assuming that the serial EEPROM values are suitable, the host only programmed the PCIBARx and PCIUBARx registers.

Note that pulling PRV\_DEV high enables Private Device operation. (Refer to Section 10.6 for further details.)

### 10.7.3 Non-Transparent Mode Address Translation

This subsection provides specific details for programming the PCI 6540 Address Translation Control registers for support of non-transparent operations, illustrating a typical sequence performed by the secondary host.

The secondary host first determines which resources to make accessible to the host on the bridge primary bus. The following combinations can be provided:

- One 32-bit I/O translated address range and one 64-bit Memory translated address range
- One 32-bit I/O translated address range and two 32-bit Memory translated address ranges
- One 32-bit memory translated address range and one 64-bit Memory translated address range
- Three 32-bit memory translated address ranges

To specify resources for the primary host, the secondary host can program the Downstream BAR 0, 1, and 2 Translation Mask registers (DWNBAR0MSK, DWNBAR1MSK, and DWNBAR2MSK; Ext:0Fh, respectively). The primary interface uses these registers to configure its BARs.

The first field in the Translation Mask register specifies the amount of address space required by the PCI 6540. The value programmed in this field is interpreted as a bit position into the corresponding BAR in primary Configuration space. The BIOS determines the amount of address space requested by the BAR by writing the value of 0Fh and reading back the register. The read value returned is ones (1) in all bit positions above the value specified in the Translation Mask register. *For example*, to request 4 KB of Address space, the BAR returns FFFF\_F000h. This is specified in the Translation Mask register by programming a value of Bh (1011b, 11d) in the MSB position of Address Mask field, which indicates a need for 11 zeros (0) in the address size, or 4 KB.

The Downstream BAR 0 Translation Mask register has a BAR Type bit (DWNBAR0MSK[6]; Ext:0Fh), which can be used to specify whether BAR 0 indicates an I/O or Memory range. This bit becomes reflected in the PCI Downstream I/O BAR 0 register (PCIBAR0[0]; Primary PCI:10h). In addition, the Prefetchable bit (DWNBAR0MSK[7]; Ext:0Fh) specifies whether the Address register points to Prefetchable Address space, and is reflected in the corresponding BAR 0 register, bits [2:1], if specified as a Memory range.

**Note:** In Transparent mode, PCIBAR0 must map into I/O space (not prefetchable) and DWNBAR0MSK[7] is **not** used. In Non-Transparent mode, DWNBAR0MSK[7] can map into either Memory or I/O space.

The Downstream BAR 1 Translation Mask register (DWNBAR1MSK; Ext:0Fh) can only configure the Downstream Memory BAR 1 register (PCIBAR1; Primary PCI:14h). PCIBAR1 can be configured as prefetchable or non-prefetchable by way of the Prefetchable bit (DWNBAR1MSK[15]; Ext:0Fh). In addition, the BAR Type bit (DWNBAR1MSK[14]; Ext:0Fh) allows configuration as a 32- or 64-bit register. If programmed as a 64-bit register, there is no need to program the BAR 2 Translation Mask register.

The Downstream BAR 2 Translation Mask register (DWNBAR2MSK; Ext:0Fh) can configure the third Base Address register (PCIBAR2; Primary PCI:18h) only as a 32-bit Base Address register, and can be selected as a prefetchable or non-prefetchable address range by way of the Prefetchable bit (DWNBAR2MSK[23]; Ext:0Fh).

After programming the Mask registers, the secondary host programs the Downstream BAR 0, 1, and 2 Translation Address registers (DWNTNBAR0; Ext:0Ch, DWNTNBAR1; Ext:0Dh, and DWNTNBAR2; Ext:0Eh, respectively). The programmed addresses are the starting address of each of the shared address spaces on the secondary interface address map.

At this point, the secondary host has completed its programming and allows the primary bus to be configured, unless P\_BOOT=0 and is indicated otherwise by the S\_PORT\_READY bit (UPSTNE [31]=0; Ext:0Bh).

The primary bus BARs can be configured by BIOS, but require software to enable translation by programming the Downstream Translation Enable register (DWNTNE; Ext:0Fh).

Alternately, these registers can be programmed into the serial EEPROM device, to be autoloaded during power-up.

# 11 TRANSACTION ORDERING

This section describes transaction ordering in Conventional PCI and PCI-X modes.

To maintain data coherency and consistency, the PCI 6540 complies with *PCI r2.3* and *PCI-X r1.0b* cross-bridge transaction ordering rules.

## 11.1 CONVENTIONAL PCI TRANSACTION ORDERING

This subsection describes the ordering rules that control Conventional PCI transaction forwarding across the PCI 6540. For a detailed discussion of transaction ordering, refer to *PCI r2.3*, Appendix E.

### 11.1.1 Transactions Governed by Ordering Rules

In Conventional PCI mode, ordering relationships are established for the following classes of transactions crossing the PCI 6540:

- **Posted Write Transactions (Comprised of Memory Write and Memory Write and Invalidate Transactions)**—Completed at the source before completing at the destination (*that is*, data is written into intermediate Data buffers before reaching the target).
- **Delayed Write Request Transactions (Comprised of I/O Write and Configuration Write Transactions)**—Terminated by Target Retry on the initiator bus and queued in the Delayed Transaction queue. A Delayed Write transaction must complete on the target bus before completing on the initiator bus.
- **Delayed Write Completion Transactions (Comprised of I/O Write and Configuration Write Transactions)**—Completed on the target bus, with the target response queued in the buffers. A Delayed Write Completion transaction proceeds in the direction opposite to that of the original Delayed Write request (*that is*, the transaction proceeds from target-to-initiator bus).
- **Delayed Read Request Transactions (Comprised of all Memory Read, I/O Read, and Configuration Read Transactions)**—Terminated by Target Retry on the initiator bus and queued in the Delayed Transaction queue.

- **Delayed Read Completion Transactions (Comprised of all Memory Read, I/O Read, and Configuration Read Transactions)**—Completed on the target bus, and the Read data was 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*, the transaction proceeds from target-to-initiator bus).

The PCI 6540 does not combine or merge Write transactions. It does not:

- **Combine separate Write transactions into a single Write transaction**—This optimization is best implemented in the originating master.
- **Merge bytes on separate Masked Write transactions to the same DWORD address**—This optimization is also best implemented in the originating master.
- **Collapse sequential Write transactions to the same address into a single Write transaction**—*PCI r2.3* does not allow combining of transactions.

### 11.1.2 General Ordering Guidelines

Conventional PCI-independent transactions on the primary and secondary buses have a relationship only when those transactions cross the PCI 6540.

The following general ordering guidelines govern transactions crossing the PCI 6540:

- 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 a Target Retry can be accepted and completed in any order with respect to other transactions terminated with a 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 completes. If more than one Delayed transaction is initiated, the initiator should repeat all the Delayed transaction requests, using a fairness algorithm. Repeating a Delayed transaction cannot be contingent on completion of another Delayed transaction; otherwise, deadlock may occur.

- Write transactions flowing in one direction have no ordering requirements with respect to Write transactions flowing in the other direction. The PCI 6540 can simultaneously accept Posted Write transactions on both interfaces, as well as simultaneously initiate Posted Write transactions on both interfaces.
- Acceptance of a Posted Memory Write transaction as a target can never be contingent on the completion of an unlocked, unposted transaction as a master. This is true of the PCI 6540 and must also be true of other bus agents; otherwise, deadlock may occur.
- PCI 6540 accepts Posted Write transactions, regardless of the state of completion of Delayed transactions being forwarded across the PCI 6540.

### 11.1.3 Ordering Rules

Table 11-1 delineates the ordering relationships of all transactions and refers, by number, to the ordering rules that follow.

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 11-1. These ordering rules apply to Posted Write transactions, Delayed Write and Read requests, and Delayed Write and Read Completion transactions crossing the PCI 6540 in the same direction. Note that delayed completion transactions cross the PCI 6540 in the direction opposite to that of the corresponding delayed requests.

1. **Posted Write**—Posted Write transactions must complete on the target bus in the order in which the transactions were received on the initiator bus.

The subsequent Posted Write transaction could 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, devices checking that flag could subsequently be using stale data.

2. **Delayed Write Request**—Delayed Write requests cannot pass previously queued Posted Write data. As in the case of Posted Memory Write transactions, the Delayed Write transaction might be setting a flag regarding data in the Posted Write transaction. If the Delayed Write request were to complete before the earlier Posted Write transaction, devices checking the flag could subsequently be using stale data.

3. **Delayed Read Request**—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 might be in the same location as the Write data; therefore, if the Read transaction were to pass the Write transaction, the read would return stale data.

4. **Delayed Write Completion**—Posted Write transactions must be provided opportunities to pass Delayed Read and Write requests and completions. Otherwise, deadlock 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 and Delayed Transaction queues.

The PCI 6540 can return Delayed Read transactions in a different order than requested if the DRT Out-of-Order Enable bit is set to 1 (MSCOPT[2]=1; PCI:46h). Requested cycles can execute out of order across the bridge, if all other ordering rules are satisfied. Therefore, if the PCI 6540 starts a Delayed transaction that is Retried by the target, the PCI 6540 can execute another transaction in the Delayed Transaction Request queue. Also, if there are Delayed Write and Read requests in the queue, and the Read Data FIFOs are full, the PCI 6540 may execute the Delayed Write request before the Delayed Read request.

On cycle completion, the PCI 6540 may complete cycles in a different order than that requested by the initiator.

5. **Delayed Read Completion**—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 bridge as the target of the Write transaction. The Posted Write transaction must complete to the target before Read data is returned to the initiator.

The Read transaction could be to a Status register of the initiator of the Posted Write data and therefore should not complete until the Write transaction is complete.



Table 11-1. Conventional PCI Transaction Ordering Summary

Pass	Posted Write	Delayed Write Request	Delayed Read Request	Delayed Write Completion	Delayed Read Completion
Posted Write	N <sup>1</sup>	Y <sup>4</sup>	Y <sup>4</sup>	Y <sup>4</sup>	Y <sup>4</sup>
Delayed Write Request	N <sup>5</sup>	Y	Y	Y	Y
Delayed Read Request	N <sup>3</sup>	Y	Y	Y	Y
Delayed Write Completion	Y	Y	Y	Y	Y
Delayed Read Completion	N <sup>2</sup>	Y	Y	Y	Y

**Note:** The superscript accompanying certain table entries refers to the five applicable ordering rules listed in Section 11.1.3. Many entries are not governed by these ordering rules; therefore, the implementation can choose whether the transactions pass each other. A “Y” indicates that transactions may be completed out of order or “pass” each other. An “N” indicates that the row transaction must not pass the column transaction.

### 11.1.4 Data Synchronization

Data synchronization refers to the relationship between interrupt signaling and data delivery. *PCI r2.3* provides the following alternative methods for synchronizing data and interrupts:

- Device signaling the interrupt performs a read of the data just written (software)
- 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

The PCI 6540 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, from the PCI 6540 to the location recently written (or some other location along the same path), or from the device driver to one of the PCI 6540 registers.

## 11.2 PCI-X TRANSACTION ORDERING

This subsection describes the ordering rules that control PCI-X transaction forwarding across the PCI 6540. For a detailed discussion of transaction ordering, refer to *PCI-X r1.0b, Use of Relaxed Ordering Appendix, Chapter 11*.

PCI-X introduces two features that affect transaction ordering and passing rules that are not present in Conventional PCI—Relaxed Ordering Attribute bit and Split transactions.

### 11.2.1 Relaxed Ordering Attribute Bit

The PCI-X Relaxed Ordering Attribute bit (AD29) may be used to allow a Memory Write transaction to pass other Memory Writes and to allow Split Read Completions to pass Memory Writes.

- If the Relaxed Ordering Attribute bit is set for a Read transaction, the completion for that transaction is allowed to pass previously Posted Memory Write transactions traveling in the direction of the completion.
- If the Relaxed Ordering Attribute bit is set for a Memory Write transaction, that transaction is allowed to pass previous Posted Memory Write transactions moving in the same direction on the host bridge.

**Note:** The PCI 6540 ignores the Relaxed Ordering Attribute bit for Memory Write transactions and maintains the order of Memory Write transactions that cross the PCI 6540.

### 11.2.2 Split Transactions

In PCI-X, Split Transaction ordering and Deadlock-Avoidance rules are almost identical to the Delayed Transaction rules in Conventional PCI (transaction order is established as the transactions complete). Table 11-2 lists the ordering requirements for all Split and Memory Write transactions (the columns represent the first of two transactions, and the rows represent the second). Table 11-3 provides a case-by-case discussion of Split transactions.

- Split Requests can be reordered with respect to other Split Requests. If an initiator requires two Split transactions to complete in order, the initiator must not issue the second request until the first Split transaction completes.
- Split Read Completions that have the same Sequence ID must remain in address order. The completer must supply the Split Read Completions on the bus, in address order, to guarantee that the requester always receives the data in its natural order. There are no ordering restrictions for Split Read Completions with different Sequence IDs.

Table 11-2. Transactions Ordering and Deadlock-Avoidance Rules

Row Pass Column?	Memory Write (Column 2)	Split Read Request (Column 3)	Split Write Request (Column 4)	Split Read Completion (Column 5)	Split Write Completion (Column 6)
Memory Write (Row A)	a) No b) Y/N	Yes	Yes	Yes	Yes
Split Read Request (Row B)	No	Y/N	Y/N	Y/N	Y/N
Split Write Request (Row C)	No	Y/N	Y/N	Y/N	Y/N
Split Read Completion (Row D)	a) No b) Y/N	Yes	Yes	a) Y/N b) No	Y/N
Split Write Completion (Row E)	Y/N	Yes	Yes	Y/N	Y/N

**Legend:**

**Yes** = Second transaction must be allowed to pass the first transaction to avoid deadlock.

**Y/N** = PCI 6540 may optionally allow the second transaction to pass the first.

**No** = Second transaction must not be allowed to pass the first transaction (to preserve strong write ordering).

**Table 11-3. Split Transactions—Case-by-Case Discussion**

Split Transaction	Comments
A2a	Unless the Relaxed Ordering Attribute bit is set, a Memory Write transaction must not pass any other Memory Writes.
A2b	If the Relaxed Ordering attribute bit is set, that Memory Write transaction is allowed to pass all other Memory Writes in the host bridge only.
A3, A4	A Memory Write transaction must be allowed to pass Split Requests to avoid deadlock.
A5, A6	A Memory Write transaction must be allowed to pass Split Completions to avoid deadlock.
B2, C2	Split Requests cannot pass a Memory Write transaction.
B3, B4	Split Requests are allowed to be blocked by, or pass, Split Completions.
C3, C4	Split transactions have the same requirements as Delayed transactions in Conventional PCI mode.
B5, B6	Split Requests are allowed to be blocked by, or pass, Split Completions.
C5, C6	Split Requests pass Split Completions. (Deadlock does not occur if Split Completions block Split Requests.)
D2a	Unless the Relaxed Ordering Attribute bit is set, Split Read Completions cannot pass a Memory Write.
D2b	If the Relaxed Ordering Attribute bit is set, the Split Read Completion is allowed to pass a previous Posted Memory Write transaction.
D3, D4	Split Read Completions must be allowed to pass all Split Requests to avoid deadlock.
D5a	Unless two Split Read Completions are part of the same Sequence, they are allowed to be blocked by, or pass, each other.
D5b	Split Read Completions with the same Sequence ID must remain in address order.
D6	Split Read Completions are allowed to be blocked by, or pass, Split Write Completions.
E2	Split Write Completions are allowed to be blocked by, or pass, Memory Write transactions moving in the opposite direction (they have no ordering relationship).
E3, E4	Split Write Completions must be allowed to pass all Split Requests to avoid deadlock.
E5, E6	Split Write Completions are allowed to be blocked by, or pass, Split Read and Write Completions.

## 12 ERROR HANDLING

This section provides detailed information regarding how the PCI 6540 manages errors. It also describes error status reporting and error operation disabling.

### 12.1 OVERVIEW

The PCI 6540 checks, forwards, and generates parity on the primary and secondary interfaces. To maintain transparency, the PCI 6540 can either forward the existing parity condition from one bus to the other, along with address and data, or regenerate the data parity on the other bus. The PCI 6540 always attempts to be transparent when reporting errors, but this is not always possible because of the presence of posted data and Delayed transactions.

To support error reporting on the PCI Bus, the PCI 6540 implements the following:

- P\_PERR#, P\_SERR#, S\_PERR#, and S\_SERR# signals
- Primary and secondary Status registers (Transparent mode—PCISR; PCI:06h and PCISSR; PCI:1Eh, Non-Transparent mode—PCISR; Primary PCI:06h, Secondary PCI:46h and PCISSR; Primary PCI:46h, Secondary PCI:06h)
- Device-specific P\_SERR# Event Disable and Status registers (Transparent mode—PSERRED; PCI:64h and PSERRSR; PCI:6Ah, Non-Transparent mode—PSERRED; PCI:96h and PSSERRSR[3:0]; PCI:98h)
- **Non-Transparent mode only**—Device-specific S\_SERR# Event Disable and Status registers (SSERRED; PCI:97h and PSSERRSR[7:4]; PCI:98h)

### 12.2 ADDRESS PARITY ERRORS

The PCI 6540 checks address parity for all bus transactions, and address and bus commands.

When the PCI 6540 detects an Address Parity error on the primary interface, the following occurs:

1. In Conventional PCI mode, if the Command register Parity Error Response Enable bit is set (Transparent mode—PCICR[6]=1; PCI:04h, Non-Transparent mode—PCICR[6]=1; Primary PCI:04h, Secondary PCI:44h), the PCI 6540 does not claim the transaction with P\_DEVSEL#.

In PCI-X mode, with a PCI-X command other than Split Completion, the PCI 6540 claims the bus as usual and immediately terminates the cycle with a Target Abort. In either case, the PCI 6540 does **not** accept the cycle in its FIFO operation.

In PCI-X mode, with a PCI-X Split Completion command, the PCI 6540 receives the cycle as if there is no Parity error.

If the Parity Error Response Enable bit is not set, the PCI 6540 proceeds as usual and accepts the transaction if the transaction is directed to, or across, the PCI 6540.

2. PCI 6540 sets the Status register Detected Parity Error bit (Transparent mode—PCISR[15]=1; PCI:06h, Non-Transparent mode—PCISR[15]=1; Primary PCI:06h, Secondary PCI:46h).
3. PCI 6540 asserts P\_SERR# and sets the Status register Signaled System Error bit (PCISR[14]=1), if the Command register P\_SERR# Enable and Parity Error Response Enable bits are set (PCICR[8, 6]=11b).

When the PCI 6540 detects an Address Parity error on the secondary interface, the following occurs:

1. In Conventional PCI mode, if the Bridge Control register Parity Error Response Enable bit is set (Transparent mode—BCNTRL[0]=1; PCI:3Eh, Non-Transparent mode—BCNTRL[0]=1; PCI:42h), the PCI 6540 does not claim the transaction with S\_DEVSEL#.

In PCI-X mode, with a PCI-X command other than Split Completion, the PCI 6540 claims the bus as usual and immediately terminates the cycle with a Target Abort. In either case, the PCI 6540 does **not** accept the cycle in its FIFO operation.

In PCI-X mode, with a PCI-X Split Completion command, the PCI 6540 receives the cycle as if there is no Parity error.

If the Parity Error Response Enable bit is not set, the PCI 6540 proceeds as usual and accepts the transaction if the transaction is directed to, or across, the PCI 6540.

2. PCI 6540 sets the Secondary Status register Detected Parity Error bit (Transparent mode—PCISSR[15]=1; PCI:1Eh, Non-Transparent mode—PCISSR[15]=1; Primary PCI:46h, Secondary PCI:06h).
3. PCI 6540 asserts S\_SERR# and sets the Status register Signaled System Error bit (PCISSR[14]=1), if the following conditions are met:
  - **Non-Transparent mode only**—Command register S\_SERR# Enable bit is set (PCISCR[8]=1; Primary PCI:44h, Secondary PCI:04h), and
  - Bridge Control register Parity Error Response Enable bit is set (BCNTRL[0]=1)

## 12.3 ATTRIBUTE PARITY ERRORS (PCI-X MODE)

PCI-X Attribute Parity errors are managed in the same way as Address Parity errors. (Refer to Section 12.2.)

## 12.4 DATA PARITY ERRORS

When forwarding transactions, the PCI 6540 attempts to pass the data parity condition from one interface to the other unchanged, whenever possible, to allow the master and target devices to manage the error condition.

The following subsections describe, for each transaction, the sequence that occurs when a Parity error is detected and the way in which the parity condition is forwarded across the bridge.

### 12.4.1 Configuration Write Transactions to Configuration Space

When the PCI 6540 detects a Data Parity error during a Type 0 Configuration Write transaction to Configuration space, the following occurs:

1. If the Command register Parity Error Response Enable bit is set (Transparent mode—PCICR[6]=1; PCI:04h, Non-Transparent mode—PCICR[6]=1; Primary PCI:04h, Secondary PCI:44h), the PCI 6540 asserts P\_PERR#. If the Parity Error Response Enable bit is not set, the PCI 6540 does **not** assert P\_PERR#. In either case, the Configuration register is written.
2. PCI 6540 sets the Status register Detected Parity Error bit (Transparent mode—PCISR[15]=1; PCI:06h, Non-Transparent mode—PCISR[15]=1; Primary PCI:06h, Secondary PCI:46h), regardless of the Parity Error Response Enable bit state (PCICR[6]=X).

### 12.4.2 Read Transactions

When the PCI 6540 detects a Parity error during a Read transaction, the target drives data and data parity, and the initiator checks parity and conditionally asserts P\_PERR# or S\_PERR#.

For downstream transactions, when the PCI 6540 detects a Read Data Parity error on the secondary bus, the PCI 6540:

1. Asserts S\_PERR# two cycles following the Data transfer, if the secondary interface Bridge Control register Parity Error Response Enable bit is set (Transparent mode—BCNTRL[0]=1; PCI:3Eh, Non-Transparent mode—BCNTRL[0]=1; PCI:42h).
2. Sets the secondary Status register Detected Parity Error bit (Transparent mode—PCISSR[15]=1; PCI:1Eh, Non-Transparent mode—PCISSR[15]=1; Primary PCI:46h, Secondary PCI:06h).
3. Sets the secondary Status register Data Parity Error Detected bit (PCISSR[8]=1), if the secondary interface Bridge Control register Parity Error Response Enable bit is set.
4. Returns the bad parity with the data to the initiator on the primary bus. If the data with the bad parity is prefetched and not read by the initiator on the primary bus, the data is discarded and data with bad parity is not returned to the initiator.
5. Completes the transaction as usual.

For upstream transactions, when the PCI 6540 detects a Read Data Parity error on the primary bus, the PCI 6540:

1. Asserts P\_PERR# two cycles following the Data transfer, if the primary interface Command register Parity Error Response Enable bit is set (PCICR[6]=1).
2. Sets the primary Status register Detected Parity Error bit (PCISR[15]=1).
3. Sets the primary Status register Data Parity Error Detected bit (PCISR[8]=1), if the primary interface Command register Parity Error Response Enable bit is set.
4. Returns the bad parity with the data to the initiator on the secondary bus. If the data with the bad parity is prefetched and not read by the initiator on the secondary bus, the data is discarded and data with bad parity is not returned to the initiator.
5. Completes the transaction as usual.

The PCI 6540 returns to the initiator the data and parity received from the target. When the initiator detects a Parity error on this Read data and is enabled to report the error, the initiator asserts its PERR# signal (which is then connected to the PCI 6540 P\_PERR# or S\_PERR# signal, depending on the bus to which the initiator is connected) two cycles after the Data transfer. It is assumed that the initiator takes responsibility for handling a Parity error condition; therefore, when the PCI 6540 detects the initiator's PERR# assertion while returning Read data to the initiator, the PCI 6540 takes no further action and completes the transaction as usual.

### 12.4.3 Delayed Write Transactions

The PCI 6540 detects a Data Parity error during a Delayed Write transaction and conditionally asserts PERR#. The PCI 6540 either passes or regenerates data parity to the target bus.

For Delayed Write transactions, a Parity error can occur:

- During the original Delayed Write Request transaction
- When the initiator repeats the Delayed Write Request transaction
- When the PCI 6540 completes the Delayed Write transaction to the target

#### 12.4.3.1 Conventional PCI Mode

In Conventional PCI mode, when a Delayed Write transaction is queued, the Address, Command, Address and Data Parity, Data, and Byte Enable bits are captured and a Target Retry is returned to the initiator. When the PCI 6540 detects a Parity error on the Write data for the initial Delayed Write Request transaction, the following occurs:

1. If the Parity Error Response Enable bit corresponding to the initiator bus is set (primary—PCICR[6]=1, secondary—BCNTRL[0]=1), the PCI 6540 asserts P\_PERR# or S\_PERR# two clocks after the data. The PCI 6540 always accepts the cycle, and can optionally pass the incorrect parity to the other bus, or regenerate the Parity bit on the other bus.
2. PCI 6540 sets the Status register Detected Parity Error bit corresponding to the initiator bus (primary—PCISR[15]=1, secondary—PCISSR[15]=1), regardless of the Parity Error Response bit state.

Following the initiating transaction (the first PCI 6540 Retry), the subsequent Data Parity error of similar transaction on the initiating bus is detected as usual; however, the Data Parity error no longer affects FIFO operation. The cycles are considered similar if they have the same Address, Command, Byte Enables and Write data. The Parity bit is not part of this “similar” detection operation. Therefore, if a Data Parity error occurs only in the Parity bit (same data as before), the cycle operates as usual. Conversely, if a Data Parity error occurs in the data segment (different data from the initiating Write data), the PCI 6540 treats the error as a new Delayed Write transaction.

#### 12.4.3.2 PCI-X Mode

In PCI-X mode, a Delayed Write transaction is queued with its Address, Command, Parity, Data, and Byte Enable bits, then returns a Split Response to the initiator. When the PCI 6540 detects a Parity error on the Write data, the following occurs:

1. If the Parity Error Response Enable bit corresponding to the initiator bus is set (primary—PCICR[6]=1, secondary—BCNTRL[0]=1), the PCI 6540 asserts P\_PERR# or S\_PERR#. The PCI 6540 always accepts the cycle, and can

optionally pass the incorrect parity to, or regenerate the Parity bit on, the other bus.

2. The PCI 6540 sets the Status register Detected Parity Error bit corresponding to the initiator bus (primary—PCISR[15]=1, secondary—PCISSR[15]=1), regardless of the Parity Error Response bit state.

#### 12.4.4 Split Completion (PCI-X Mode)

When detecting a Data Parity error on the originating bus for a Split Completion other than a Split Completion Message, the PCI 6540 asserts P\_PERR# or S\_PERR# and sets the appropriate Error Status bit for that interface. The PCI 6540 “drives bad parity” when it forwards the Split Completion.

When the PCI 6540 detects a Data Parity error on the originating bus for a Split Completion Message, the following occurs:

- **PCI-X-to-PCI mode**—Parity error is not considered in the FIFO operation. If the Split Completion Error bit is set to 1 (AD30=1), the PCI 6540 signals a Target Abort; otherwise, it is considered to be a normal completion.
- **PCI-X-to-PCI-X mode**—PCI 6540 forwards the exact message with the incorrect Parity bit to the other bus.

#### 12.4.5 Posted Write Transactions

During downstream Posted Write transactions, when the PCI 6540, responding as a target, detects a Data Parity error on the initiator (primary) bus, the PCI 6540:

1. Asserts P\_PERR# two cycles after the Data transfer, if the primary interface Command register Parity Error Response Enable bit is set (PCICR[6]=1).
2. Sets the primary interface Status register Detected Parity Error bit (PCISR[15]=1).
3. Captures and forwards the bad parity condition to the secondary bus.
4. Completes the transaction as usual.

Similarly, during upstream Posted Write transactions, when the PCI 6540, responding as a target, detects a Data Parity error on the initiator (secondary) bus, the PCI 6540:

1. Asserts S\_PERR# two cycles after the Data transfer, if the secondary interface Bridge Control register Parity Error Response Enable bit is set (BCNTRL[0]=1).
2. Sets the secondary interface Status register Detected Parity Error bit (PCISSR[15]=1).
3. Captures and forwards the bad parity condition to the primary bus.
4. Completes the transaction as usual.

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 PCI 6540:

1. Sets the secondary Status register Data Parity Error Detected bit (PCISSR[8]=1), if the secondary interface Bridge Control register Parity Error Response Enable bit is set (BCNTRL[0]=1).
2. Asserts P\_SERR# and sets the Status register Signaled System Error bit (PCISR[14]=1), if the following conditions are met:
  - Primary interface Command register P\_SERR# Enable and Parity Error Response Enable bits are set (PCICR[8, 6]=11b), and
  - Device-specific P\_SERR# Disable bit for Posted Write Parity errors is not set (PSERRED[1]=0), and
  - Secondary interface Bridge Control register Parity Error Response Enable bit is set (BCNTRL[0]=1), and
  - PCI 6540 did not detect the Parity error on the initiator (primary) 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 PCI 6540:

1. Sets the Status register Data Parity Error Detected bit (PCISR[8]=1), if the primary interface Command register Parity Error Response Enable bit is set (PCICR[6]=1).
2. Asserts P\_SERR# and sets the Status register Signaled System Error bit (PCISR[14]=1), if the following conditions are met:
  - Primary interface Command register P\_SERR# Enable and Parity Error Response Enable bits are set (PCICR[8, 6]=11b), and
  - Secondary interface Bridge Control register Parity Error Response Enable bit is set (BCNTRL[0]=1), and
  - PCI 6540 did not detect the Parity error on the initiator (secondary) bus (*that is*, the Parity error was not forwarded from the secondary bus)

P\_SERR# assertion signals the Parity error condition when the initiator is not sent information about an error having occurred. Because the data is delivered with no errors, there is no other way to signal this information to the initiator.

If a Parity error is forwarded from the initiator bus to the target bus, P\_SERR# is **not** asserted.

## 12.5 DATA PARITY ERROR REPORTING SUMMARY

In the previous subsections, the PCI 6540 responses to Data Parity errors are presented according to transaction type in progress. This subsection organizes the PCI 6540 responses to Data Parity errors according to the Status bits set by the PCI 6540 and the signals asserted.

Table 12-1 delineates the primary interface Status register Detected Parity Error bit status. This bit is set when the PCI 6540 detects a Parity error on the primary interface.

Table 12-2 delineates the secondary interface Status register Detected Parity Error bit status. This bit is set when the PCI 6540 detects a Parity error on the secondary interface.

Table 12-3 delineates the primary interface Status register Data Parity Error Detected bit status. This bit is set under the following conditions:

- PCI 6540 must be a master on the primary bus, and
- Primary interface Command register Parity Error Response Enable bit must be set (PCICR[6]=1), and
- P\_PERR# is detected asserted, or a Parity error is detected on the primary bus

Table 12-4 delineates the secondary interface Status register Data Parity Error Detected bit status. This bit is set under the following conditions:

- PCI 6540 must be a master on the secondary bus, and
- Secondary interface Bridge Control register Parity Error Response Enable bit must be set (BCNTRL[0]=1), and
- S\_PERR# is detected asserted, or a Parity error is detected on the secondary bus

Table 12-5 delineates P\_PERR# assertion. This signal is set under the following conditions:

- PCI 6540 is either the target of a Write transaction or the initiator of a Read transaction on the primary bus, and
- Primary interface Command register Parity Error Response Enable bit must be set (PCICR[6]=1), and
- PCI 6540 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 12-6 delineates S\_PERR# assertion. This signal is set under the following conditions:

- PCI 6540 is either the target of a Write transaction or the initiator of a Read transaction on the secondary bus, and
- Secondary interface Bridge Control register Parity Error Response Enable bit must be set (BCNTRL[0]=1), and
- PCI 6540 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 12-7 delineates P\_SERR# assertion. This signal is set under the following conditions:

- Command register P\_SERR# Enable and Parity Error Response Enable bits must be set (PCICR[8, 6]=11b),
- Bridge Control register Parity Error Response Enable bit must be set (BCNTRL[0]=1), and
- PCI 6540 detects S\_PERR# asserted on a downstream Posted Write transaction, or P\_PERR# asserted on an upstream Posted Write transaction, and
- PCI 6540 did not detect the Parity error as a target of the Posted Write transaction

## 12.6 SYSTEM ERROR (SERR#) REPORTING

In this data book, when P\_SERR# assertion is discussed, the following conditions are assumed:

- For the PCI 6540 to assert P\_SERR#, the Command register P\_SERR# Enable bit must be set (PCICR[8]=1)
- When the PCI 6540 asserts P\_SERR#, the PCI 6540 must also set the Status register Signaled System Error bit (PCISR[14]=1)

In compliance with *P-to-P Bridge r1.1*, the PCI 6540 asserts P\_SERR# when it detects the S\_SERR# input asserted and the Bridge Control register S\_SERR# Enable bit is set (BCNTRL[1]=1). In addition, the PCI 6540 also sets the secondary Status register Signaled System Error bit (PCISSR[14]=1).

The PCI 6540 also conditionally asserts P\_SERR# for the following conditions:

- Master Abort detected during Posted Write transaction (on the secondary bus)
- Target Abort detected during Posted Write transaction (on the secondary bus)
- Posted Write data discarded after  $2^{24}$  delivery attempts ( $2^{24}$  Target Retries received)
- S\_PERR# reported on the target bus during a Posted Write transaction (refer to Section 12.5)
- Delayed Write data discarded after  $2^{24}$  delivery attempts ( $2^{24}$  Target Retries received)
- Delayed Read data cannot be transferred from the 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 P\_SERR# assertion.

Most of these events have additional device-specific Disable bits in the P\_SERR# Event Disable register that can mask P\_SERR# assertion for specific events. The Master Timeout condition has P\_SERR# and S\_SERR# Enable bits for that event in the Bridge Control register (BCNTRL[12:11]), and therefore does not have a device-specific Disable bit.

Table 12-1. Primary Interface Detected Parity Error Bit Status

Primary Detected Parity Error Bit (PCISR[15])	Transaction Type	Direction	Bus on which Error Detected	Primary Parity Error Response Enable Bit (PCICR[6])	Secondary Parity Error Response Enable Bit (BCNTRL[0])
0	Read	Downstream	Primary	x	x
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

**Note:** x = Don't care.

Table 12-2. Secondary Interface Detected Parity Error Bit Status

Secondary Detected Parity Error Bit (PCISSR[15])	Transaction Type	Direction	Bus on which Error Detected	Primary Parity Error Response Enable Bit (PCICR[6])	Secondary Parity Error Response Enable Bit (BCNTRL[0])
0	Read	Downstream	Primary	x	x
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

Table 12-3. Primary Interface Data Parity Error Detected Bit Status

Primary Data Parity Error Detected Bit (PCISR[8])	Transaction Type	Direction	Bus on which Error Detected	Primary Parity Error Response Enable Bit (PCICR[6])	Secondary Parity Error Response Enable Bit (BCNTRL[0])
0	Read	Downstream	Primary	x	x
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

**Note:** x = Don't care.

Table 12-4. Secondary Interface Data Parity Error Detected Bit Status

Secondary Data Parity Error Detected Bit (PCISSR[8])	Transaction Type	Direction	Bus on which Error Detected	Primary Parity Error Response Enable Bit (PCICR[6])	Secondary Parity Error Response Enable Bit (BCNTRL[0])
0	Read	Downstream	Primary	x	x
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

Table 12-5. P\_PERR# Assertion

P_PERR#	Transaction Type	Direction	Bus on which Error Detected	Primary Parity Error Response Enable Bit (PCICR[6])	Secondary Parity Error Response Enable Bit (BCNTRL[0])
1 (De-asserted)	Read	Downstream	Primary	x	x
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*	Delayed Write	Downstream	Secondary	1	1
1	Delayed Write	Upstream	Primary	x	x
1	Delayed Write	Upstream	Secondary	x	x

**Notes:** x = Don't care.

\* Parity error detected on the target (secondary) bus, but not on the initiator (primary) bus.

Table 12-6. S\_PERR# Assertion

S_PERR#	Transaction Type	Direction	Bus on which Error Detected	Primary Parity Error Response Enable Bit (PCICR[6])	Secondary Parity Error Response Enable Bit (BCNTRL[0])
1 (De-asserted)	Read	Downstream	Primary	x	x
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*	Delayed Write	Upstream	Primary	1	1
0	Delayed Write	Upstream	Secondary	x	1



Table 12-7. P\_SERR# for Data Parity Error Assertion

P_PERR#	Transaction Type	Direction	Bus on which Error Detected	Primary Parity Error Response Enable Bit (PCICR[6])	Secondary Parity Error Response Enable Bit (BCNTRL[0])
1 (De-asserted)	Read	Downstream	Primary	x	x
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* (Asserted)	Posted Write	Downstream	Secondary	1	1
0**	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

**Notes:** x = Don't care.

\* Parity error detected on the target (secondary) bus, but not on the initiator (primary) bus.

\*\* Parity error detected on the target (primary) bus, but not on the initiator (secondary) bus.



## 13 EXCLUSIVE ACCESS

This section describes P\_LOCK# and S\_LOCK# signal use to implement exclusive access to a target for transactions that cross the PCI 6540, including concurrent locks, and acquiring and ending exclusive access.

### 13.1 CONCURRENT LOCKS

The primary and secondary bus Lock mechanisms concurrently operate, *except* when a Locked transaction is crossing the PCI 6540. A primary master can lock a primary target without affecting the lock status on the secondary bus, and vice versa. This means that a primary master can lock a primary target concurrent with a secondary master locking a secondary target.

### 13.2 ACQUIRING EXCLUSIVE ACCESS ACROSS PCI 6540

For a PCI Bus, before acquiring access to the P\_LOCK# and/or S\_LOCK# signal and starting a series of Locked transactions, the initiator must first check that both of the following conditions are met:

- PCI Bus is idle, and
- P\_LOCK# and/or S\_LOCK# is de-asserted

The initiator leaves P\_LOCK# and/or S\_LOCK# de-asserted during the Address phase and asserts P\_LOCK# and/or S\_LOCK# one Clock cycle later. Target lock is achieved after the target completes a Data transfer.

Locked transactions can cross the PCI 6540 in the downstream and upstream directions, from the primary-to-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 bus. When the PCI 6540 detects an initial Locked transaction on the primary bus that is intended for a target on the secondary bus, the PCI 6540 samples the Address, Transaction Type, Byte Enable, and Parity bits, and the S\_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 nevertheless posted. Memory Read transactions that are part of the Locked transaction sequence are not prefetched.

When the Locked Delayed Read request is queued, the PCI 6540 does not queue further transactions until the locked sequence is complete. The PCI 6540 signals a Target Retry to all transactions initiated subsequent to the Locked Read transaction that are intended for targets on the opposite side of the PCI 6540. The PCI 6540 allows transactions queued before the Locked transaction to complete before initiating the Locked transaction.

When the Locked Delayed Read request moves to the head of the Delayed Transaction queue, the PCI 6540 initiates the request as a Locked Read transaction by de-asserting S\_LOCK# on the target bus during the first Address phase, then asserting S\_LOCK# one cycle later. If S\_LOCK# was previously asserted (used by another initiator), the PCI 6540 waits to request access to the secondary bus until S\_LOCK# is sampled de-asserted when the target bus is idle. Note that the existing lock on the target bus did not cross the PCI 6540; otherwise, the pending queued Locked transaction would not have queued. When the PCI 6540 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, Byte Enable and Parity bits, the PCI 6540 transfers the Read data back to the initiator, and the lock is also established on the primary bus.

For the PCI 6540 to recognize and respond to the initiator, the initiator's subsequent attempts of the Read transaction must use the Locked-transaction sequence (de-assert P\_LOCK# during the Address phase, then re-assert P\_LOCK# one cycle later). If the P\_LOCK# sequence is not used in subsequent attempts, a Master Timeout condition may result. When a Master Timeout condition occurs, P\_SERR# is conditionally asserted, the Read data and queued Read transaction are discarded, and S\_LOCK# is de-asserted on the target bus.

After the intended target is locked, subsequent Locked transactions initiated on the initiator bus that are forwarded by the PCI 6540 are driven as Locked transactions on the target bus.

When the PCI 6540 receives a Master or Target Abort in response to the Delayed Locked Read transaction, this status is passed back to the initiator, and no locks are established on the initiator or target bus. The PCI 6540 resumes forwarding Unlocked transactions in both directions.

### 13.3 ENDING EXCLUSIVE ACCESS

After the lock is acquired on the initiator and target buses, the PCI 6540 must maintain the lock on the target bus for 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 P\_LOCK# and/or S\_LOCK# signal status.

An established target lock is maintained until the initiator relinquishes the lock. The PCI 6540 does not recognize whether the current transaction is the last one in a sequence of Locked transactions until the initiator de-asserts P\_LOCK# and/or S\_LOCK# at the end of the transaction.

When the last Locked transaction is a Delayed transaction, the PCI 6540 previously completed the transaction on the secondary bus. In this case, when the PCI 6540 detects that the initiator has relinquished the P\_LOCK# and/or S\_LOCK# signal by sampling the signal de-asserted while P\_FRAME# or S\_FRAME# is de-asserted, the PCI 6540 de-asserts P\_LOCK# and/or S\_LOCK# on the target bus when possible. Because of this behavior, P\_LOCK# and/or S\_LOCK# may not be de-asserted until several cycles after the last Locked transaction completes on the target bus. After de-asserting P\_LOCK# and/or S\_LOCK# to indicate the end of a sequence of Locked transactions, the PCI 6540 resumes forwarding of Unlocked transactions.

When the last Locked transaction is a Posted Write transaction, the PCI 6540 de-asserts P\_LOCK# and/or S\_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 the PCI 6540 receives a Master or Target Abort in response to a Locked Delayed transaction, the PCI 6540 returns a Master or Target Abort when the initiator repeats the Locked transaction. The initiator must then de-assert P\_LOCK# and/or S\_LOCK# at the end of the transaction. The PCI 6540 sets the appropriate Status bits, flagging the abnormal Target Termination condition. Normal forwarding of unlocked posted and Delayed transactions is resumed.

When the PCI 6540 receives a Master or Target Abort in response to a Locked Posted Write transaction, the PCI 6540 cannot pass back that status to the initiator. The PCI 6540 asserts P\_SERR# on the initiator bus when a Master or Target Abort is received during a Locked Posted Write transaction, if the Command register P\_SERR# Enable bit is set (Transparent mode—PCICR[8]=1; PCI:04h, Non-Transparent mode—PCICR[8]=1; Primary PCI:04h, Secondary PCI:44h). P\_SERR# is asserted for the Master Abort condition if the Bridge Control register Master Abort Mode bit is set (Transparent mode—BCNTRL[5]=1; PCI:3Eh, Non-Transparent mode—BCNTRL[5]=1; PCI:42h).

**Note:** The PCI 6540 has an option to ignore the Lock protocol, by clearing the Secondary and/or Primary Lock Enable bits (MSCOPT[14:13]=00b; PCI:46h).

## 14 PCI BUS ARBITRATION

This section describes primary and secondary bus arbitration and bus parking.

### 14.1 OVERVIEW

The PCI 6540 must arbitrate for use of the secondary bus when forwarding downstream transactions, and for the primary bus when forwarding upstream transactions. The primary bus Arbiter is external to the PCI 6540 (typically located on the motherboard). For the secondary PCI Bus, the PCI 6540 has a built-in Internal Arbiter. If required, the Internal Arbiter can be disabled, allowing use of an External Arbiter on the secondary bus.

### 14.2 PRIMARY PCI BUS ARBITRATION

The PCI 6540 uses a Request output pin, P\_REQ#, and a Grant input pin, P\_GNT#, for primary PCI Bus arbitration. The PCI 6540 asserts P\_REQ# when forwarding transactions upstream (*that is*, when operating as an initiator on the primary PCI Bus). When there are one or more pending transactions in the upstream direction queues—Posted Write data or Delayed transaction requests—the PCI 6540 maintains P\_REQ# assertion. However, if a Target Retry, Disconnect, or Abort is received in response to a PCI 6540-initiated transaction on the primary PCI Bus, the PCI 6540 de-asserts P\_REQ# for two PCI Clock cycles. For all cycles passing through the bridge, P\_REQ# is not asserted until the complete transaction request is queued.

If the PCI 6540 asserts P\_REQ# and the primary bus External Arbiter asserts P\_GNT# to grant the bus to the PCI 6540, the PCI 6540 initiates a transaction on the primary bus on the next PCI Clock cycle.

If the primary bus External Arbiter asserts PCI 6540 P\_GNT# when P\_REQ# is not asserted, the PCI 6540 parks P\_ADx, P\_CBEx#, P\_PAR, and P\_PAR64 by driving these signals to valid logic levels. If the primary bus is parked on the PCI 6540 and the PCI 6540 has a transaction to initiate on the primary bus, the PCI 6540 initiates the transaction if P\_GNT# remained asserted during the cycle prior to the start of the transfer.

### 14.3 SECONDARY PCI BUS ARBITRATION

The PCI 6540 implements a secondary PCI Bus Internal Arbiter, which supports up to eight external bus masters in addition to the PCI 6540. If required, the Internal Arbiter can be disabled, allowing use of an External Arbiter for secondary bus arbitration.

#### 14.3.1 Secondary Bus Arbitration Using Internal Arbiter

To use the Internal Arbiter, the secondary bus Arbiter Enable pin, S\_CFN#, must be tied low. The PCI 6540 has eight secondary bus Request input pins, (S\_REQ[7:0]#) and eight secondary bus Grant output pins (S\_GNT[7:0]#) to support external secondary bus masters. If S\_CFN# is high, the secondary bus Request and Grant signals are internally connected to the Arbiter and the secondary bus External Arbiter connections are disabled.

The PCI 6540 uses a two-level arbitration scheme, whereby arbitration is divided into two groups—high- and low-priority. The low-priority group represents a single entry in the high-priority group. Therefore, if the high-priority group consists of  $n$  masters, the highest priority is assigned to the low-priority group at least once every  $n+1$  transactions. Priority changes evenly among the low-priority group. Therefore, assuming all masters request the bus, members of the high-priority group are 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 a high- or low-priority group, through the Arbiter Control register (Transparent mode—ACNTRL; PCI:42h, Non-Transparent mode—ACNTRL; PCI:DAh).

Each group can be programmed to use a rotating or fixed priority scheme, through the Internal Arbiter Control register Group Fixed Arbitration bits (IACNTRL[2, 0]; PCI:50h).

### 14.3.2 Rotating Priority Scheme

The secondary Arbiter supports a programmable two-level rotating algorithm that cares for the eight request/grant pairs to control up to eight external bus masters. In addition, there is a request/grant pair internal to the PCI 6540, which allows the device to request and be granted access to the secondary bus. Figure 14-1 is an example of the Internal Arbiter wherein four masters, including the PCI 6540, are in the high-priority group, and five 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 way (the PCI 6540 is denoted as *B*; high-priority members are provided in *italic type*, and low-priority members in **boldface type**):

*B*, *m0*, *m1*, *m2*, **m3**, *B*, *m0*, *m1*, *m2*, **m4**,  
*B*, *m0*, *m1*, *m2*, **m5**, and so forth

If all masters are assigned to one group, the algorithm defaults to a rotating priority among all masters. After reset, all external masters are assigned to the low-priority group, and the PCI 6540 to the high-priority group. Therefore, by default, the PCI 6540 receives highest priority on the secondary bus every other transaction and priority rotates evenly among the other masters.

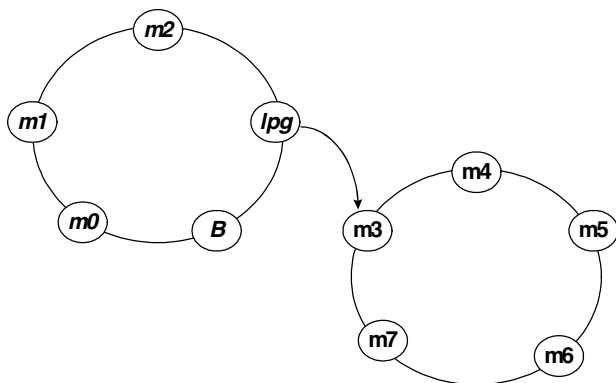


Figure 14-1. Secondary Bus Arbiter Example

**Note:** In Figure 14-1, "lpg" denotes "low-priority group."

Priorities are re-evaluated when 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 asserted. If a grant for a particular request is asserted, and a higher priority request subsequently asserts, the Arbiter de-asserts the asserted Grant signal and asserts the Grant signal corresponding to the new higher priority request on the next PCI Clock cycle. When priorities are re-evaluated, 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 within its group. Priority is also re-evaluated if the requesting agent de-asserts its request without generating cycles while the request was granted.

If the PCI 6540 detects that an initiator has failed to assert S\_FRAME# after 16 cycles of grant assertion and a secondary bus idle condition, the Arbiter re-evaluates grant assignment.

### 14.3.3 Fixed Priority Scheme

The PCI 6540 also supports a fixed priority scheme within the two priority groups. In this case, the Internal Arbiter Control register controls whether the high- or low-priority group uses the fixed or rotating priority scheme (IACNTRL[2, 0]; PCI:50h). If using a fixed priority scheme, a master within the group is assigned the highest priority within its group, and an option is set to control the priority of other masters relative to the highest priority master. This is controlled through the Internal Arbiter Control register Highest Priority Master and Group Arbitration Order bits (IACNTRL[11:4, 3, 1]; PCI:50h).

Using the example provided in Figure 14-1, but with the groups at fixed priority, suppose that:

- Master 7 has the highest priority of the low-priority group (IACNTRL[7:4]=0111b)
- PCI 6540 has the highest priority of the high-priority group (IACNTRL[11:8]=1000b)
- Priority decreases in ascending order of masters for both groups (IACNTRL[3, 1]=00b)

The order of priority with the highest first is as follows:

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

If IACNTRL[3,1]=11b, priority increases with ascending order of bus master and the order becomes:

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

Take care when using fixed arbitration in the low-priority group. As previously noted, the low-priority group receives the grant only when there are no high-priority group requests. When the Arbiter switches to the low-priority group, the highest priority master requesting the bus within that group receives the grant. If there are several requests issued by the high-priority group members and the high-priority master in the low-priority group, then lower priority devices in the low-priority group may have to wait before receiving the grant.

To prevent bus contention, if the secondary PCI Bus is idle, the Arbiter waits at least one Clock cycle between the S\_GNTx# de-assertion and assertion of the next S\_REQx#. If the secondary PCI Bus is busy (*that is*, S\_FRAME# or S\_IRDY# is asserted) when another bus master requests the bus, the Arbiter can de-assert one grant and assert the next grant during the same PCI Clock cycle.

### 14.3.4 Secondary Bus Arbitration Using External Arbiter

The Internal Arbiter can be disabled by pulling the secondary bus Central Function Control pin (S\_CFN#) high. An External Arbiter must be used if more than one bus master is required to initiate cycles on the secondary bus.

When S\_CFN# is tied high, the PCI 6540 reconfigures two pins to be external Request and Grant pins. The S\_REQ0# pin is reconfigured to be the external request output from the PCI 6540 and is used by the PCI 6540 to request the secondary bus. The S\_GNT0# pin is reconfigured to be the PCI 6540 external Grant input from the External Arbiter.

If the PCI 6540 requests the secondary PCI Bus (S\_REQ0# asserted) and the External Arbiter grants the bus to the PCI 6540 (S\_GNT0# asserted), the PCI 6540 initiates a transaction on the secondary bus one Clock cycle later.

If the secondary bus External Arbiter asserts S\_GNT0# when S\_REQ0# is not asserted, the PCI 6540 parks S\_ADx, S\_CBEx#, S\_PAR, and S\_PAR64 by driving these signals to valid logic levels.

When using an External Arbiter, the unused secondary bus Grant outputs (S\_GNT[7:0]#) are driven high. Unused secondary bus Request inputs (S\_REQ[7:0]#) must be pulled high.

## 14.4 ARBITRATION BUS PARKING

Bus parking refers to driving the ADx, CBEx#, PAR, and PAR64 lines to a known value while the bus is idle. The PCI Bus is parked on the PCI 6540 primary or secondary bus when either or both buses are idle. Bus parking occurs when the bus grant to the PCI 6540 on the parked bus is being asserted, and the PCI 6540 request for that bus is not asserted. The ADx and CBEx# signals are first driven low, then the PAR signal is driven low one cycle later.

When the GNT# signal for the parked bus is de-asserted, the PCI 6540 places the ADx, CBEx#, PAR, and PAR64 signals into a high-impedance state on the next PCI clock cycle. If the PCI 6540 is parking and wants to initiate a transaction on that bus, the PCI 6540 can start the transaction on the next PCI Clock cycle by asserting FRAME# if GNT# remains asserted.

### 14.4.1 Software Controlled PCI 64-Bit Extension Signals Parking

By reading the input status of DEV64#, U\_MODE, P\_BOOT, and TRANS#, software can determine which PCI 6540 port is interfaced to a backplane and whether it can perform 64-bit transactions.

If only 32-bit transactions can be used, then software can program the PCI 6540 to drive the unused 64-bit extension signals to 0. This is an optional mechanism for use in the event that external pull-up resistors are not desirable (which may be the case in high-speed applications) and prevents the 64-bit extension signals from floating.

The control bits are *primary 64-bit extension signals park* and *secondary 64-bit extension signals park*, located in HSSRRC[2:1]; PCI:9Ch.





## 15 GPIO INTERFACE

This section describes the GPIO interface pins and control registers.

### 15.1 GPIO INTERFACE PINS

The PCI 6540 provides 16, general-purpose I/O (GPIO) interface pins. (Refer to Table 15-1 on page 15-2) During normal operation, the Configuration registers control the GPIO interface. In addition, the GPIO pins can be used for the following:

- 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
- In Non-Transparent mode, the GPIO[14, 4] and GPIO[15, 5] pins can be used as an interrupt for communication between the primary and secondary interfaces, respectively

The GPIO[7:0] pins have weak internal pull-up resistors. External pull-up or pull-down resistors are recommended.

### 15.2 GPIO CONTROL REGISTERS

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

- Output Enable (GPIOOE)
- Output Data (GPIOOD)
- Input Data (GPIOID)

The GPIO[7:4] and GPIO[3:0] Configuration registers consist of five 8-bit fields:

- Output Enable Write 1 to Set (GPIOOE<sub>x</sub>[7:4])
- Output Enable Write 1 to Clear (GPIOOE<sub>x</sub>[3:0])
- Output Data Write 1 to Set (GPIOOD<sub>x</sub>[7:4])
- Output Data Write 1 to Clear (GPIOOD<sub>x</sub>[3:0])
- Input Data (GPIOID<sub>x</sub>[7:4])

The Output Enable fields control whether the GPIO signals are inputs or outputs. Each signal is independently controlled by a bit in each Output Enable 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 placed into a high-impedance state, and the pin is input only. Writing zeros (0) to these registers has no effect. The reset state for these signals is input only.

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 (0) to these registers has no effect. The value written to the Output Data register is driven only when the GPIO signal is configured as output. A Type 0 Configuration Write operation is used to program these registers. The reset value for the output is 0.

The Input Data field is Read-Only and reflects the current value of the GPIO pins. A Type 0 Configuration Read operation to the Input Data register returns the values of these pins. GPIO[7:0] pins can be read at any time, whether configured as input only or bi-directional.

The GPIO[15:8] Configuration registers consist of three, 8-bit fields:

- Output Data (Write 0 or 1)
- Output Enable (Write 0 or 1 to Disable or Enable)
- Input Data

The GPIO[15:8] Configuration register Output Data fields are written directly as 0 or 1. Likewise, the Output Enable fields are directly written, with a value of 1 enabling output. A Type 0 Configuration Read to the Input Data register returns the value of the GPIO[15:8] pins. During power-up, PWRGD must be a sharp rising signal to latch the status of the GPIO[15:8] pins.

Table 15-1. GPIO Pin Alternate Functions

GPIO Pin	Alternate Function
GPIO0—Pull-up	Functions as Secondary Bus Clock Mask Shift register clock output when P_RSTIN# is asserted.
GPIO1—Pull-up	No alternate function.
GPIO2 —Pull-up	Functions as Shift/Load Control Output to Shift register when P_RSTIN# is asserted.
GPIO3 —Pull-up	No alternate function.
GPIO4 —Pull-up	If Non-Transparent mode is enabled, can be used as an active low-level triggered external interrupt input to trigger P_INTA#.
GPIO5 —Pull-up	If Non-Transparent mode is enabled, can be used as an active low-level triggered external interrupt input to trigger S_INTA#.
GPIO6 —Pull-up	No alternate function.
GPIO7 —Pull-up	No alternate function.
GPIO8	Status latched during Power-up PWRGD reset.
GPIO9	Status latched during Power-up PWRGD reset.
GPIO10	Status latched during Power-up PWRGD reset.
GPIO11	Status latched during Power-up PWRGD reset.
GPIO12	Status latched during Power-up PWRGD reset.
GPIO13	Status latched during Power-up PWRGD reset. Pull low for 133 MHz Transparent mode operations. Pull high for 100 MHz Transparent mode operations. The initial state is used to determine the maximum allowable PCI-X speed status value, reported in the PCI-X Secondary Status register (PCIXSSR; PCI:F2h).
GPIO14	Status latched during power-up PWRGD reset. If Non-Transparent mode is enabled, this input can be used as an active low-level triggered external interrupt input to trigger P_INTA#.
GPIO15	Status latched during power-up PWRGD reset. If Non-Transparent mode is enabled, this input can be used as an active low-level triggered external interrupt input to trigger S_INTA#.

# 16 SUPPORTED COMMANDS

This section discusses the PCI 6540 Conventional PCI and PCI-X command set.

## 16.1 PRIMARY INTERFACE COMMAND SET

Table 16-1 delineates the PCI 6540 primary interface Conventional PCI and PCI-X command set.

**Table 16-1. Primary Interface Supported Commands**

P_CBE[3:0]#	Command		Support
	Conventional PCI	PCI-X	
0000b	Interrupt Acknowledge	Interrupt Acknowledge	<b>Not Supported.</b>
0001b	Special Cycle	Special Cycle	<b>Not Supported.</b>
0010b	I/O Read	I/O Read	If the address is within pass-through I/O range, the transaction is claimed and passed through. If the address points to an I/O-mapped internal bridge register, the transaction is claimed. Otherwise, the transaction is ignored.
0011b	I/O Write	I/O Write	Same as I/O Read (P_CBE[3:0]#=0010b).
0100b	<b>Reserved</b>	<b>Reserved</b>	—
0101b	<b>Reserved</b>	<b>Reserved</b>	—
0110b	Memory Read	Memory Read DWORD	If the address is within pass-through Memory range, the transaction is claimed and passed through. If the address points to a memory-mapped internal bridge register, the transaction is claimed. Otherwise, transaction is ignored.
0111b	Memory Write	Memory Write	Same as a Memory Read (P_CBE[3:0]#=0110b).
1000b	<b>Reserved</b>	Alias to Memory Read Block	Conventional PCI— <b>Not Supported.</b> PCI-X—Treated as a Memory Read Block (P_CBE[3:0]#=1110b).
1001b	<b>Reserved</b>	Alias to Memory Write Block	Conventional PCI— <b>Not Supported.</b> PCI-X—Treated as a Memory Write Block (P_CBE[3:0]#=1111b).

Table 16-1. Primary Interface Supported Commands (Continued)

P_CBE[3:0]#	Command		Support
	Conventional PCI	PCI-X	
1010b	Configuration Read	Configuration Read	<p>Type 0 Configuration Read, claimed if the P_IDSEL line is asserted; otherwise, the read is ignored. If claimed, the target internal register(s) read. Never passed through.</p> <p>Type 1 Configuration Read, claimed if the P_IDSEL line is asserted; otherwise, the read is ignored. If the target bus is the bridge's secondary bus, the transaction is claimed and passed through as a Type 0 Configuration Read.</p> <p>If the target bus is a subordinate bus that exists behind the bridge (but not equal to the secondary bus), the transaction is claimed and passed through as a Type 1 Configuration Read.</p>
1011b	Configuration Write	Configuration Write	<p>Type 0 Configuration Write, same as Configuration Read (P_CBE[3:0]#=1010b).</p> <p>Type 1 Configuration Write (not Special Cycle request), same as Configuration Read (P_CBE[3:0]#=1010b).</p> <p>Configuration Write as Special Cycle request (Device = 1Fh, Function = 7h).</p> <p>If the target bus is the bridge's secondary bus, the transaction is claimed and passed through as a Special Cycle.</p> <p>If the target bus is a subordinate bus that exists behind the bridge (but not equal to the secondary bus), the transaction is claimed and passed through unchanged as a Type 1 Configuration Write.</p>
1100b	Memory Read Multiple	Split Completion	<p>Conventional PCI—Treated as a Memory Read (P_CBE[3:0]#=0110b).</p> <p>PCI-X—Split Completion.</p>
1101b	DAC	DAC	Lower 32 bits of the address driven out on P_AD[31:0], followed by the upper 32 bits.
1110b	Memory Read Line	Memory Read Block	<p>Conventional PCI—Treated as a Memory Read (P_CBE[3:0]#=0110b).</p> <p>PCI-X—Memory Read Block.</p>
1111b	Memory Write and Invalidate	Memory Write Block	<p>Conventional PCI—Treated as a Memory Write.</p> <p>PCI-X—Memory Write Block.</p>

## 16.2 SECONDARY INTERFACE COMMAND SET

Table 16-2 delineates the PCI 6540 secondary interface Conventional PCI and PCI-X command set.

**Table 16-2. Secondary Interface Supported Commands**

S_CBE[3:0]#	Command		Support
	Conventional PCI	PCI-X	
0000b	Interrupt Acknowledge	Interrupt Acknowledge	<b>Not Supported.</b>
0001b	Special Cycle	Special Cycle	<b>Not Supported.</b>
0010b	I/O Read	I/O Read	If the address is within pass-through I/O range, the transaction is claimed and passed through. If the address points to an I/O-mapped internal bridge register, the transaction is claimed. Otherwise, transaction is ignored.
0011b	I/O Write	I/O Write	Same as I/O Read (S_CBE[3:0]#=0010b).
0100b	<b>Reserved</b>	<b>Reserved</b>	—
0101b	<b>Reserved</b>	<b>Reserved</b>	—
0110b	Memory Read	Memory Read DWORD	If the address is within pass-through Memory range, the transaction is claimed and passed through. If the address points to a memory-mapped internal bridge register, the transaction is claimed. Otherwise, transaction is ignored.
0111b	Memory Write	Memory Write	Same as a Memory Read (S_CBE[3:0]#=0110b).
1000b	<b>Reserved</b>	Alias to Memory Read Block	Conventional PCI— <b>Not Supported.</b> PCI-X—Treated as a Memory Read Block (S_CBE[3:0]#=1110b).
1001b	<b>Reserved</b>	Alias to Memory Write Block	Conventional PCI— <b>Not Supported.</b> PCI-X—Treated as a Memory Write Block (S_CBE[3:0]#=1111b).
1010b	Configuration Read	Configuration Read	Upstream Configuration Read cycles. <b>Not Supported.</b>
1011b	Configuration Write	Configuration Write	Type 0 Configuration Write. <b>Not Supported.</b> Type 1 Configuration Write (not a Special Cycle request). <b>Not Supported.</b> Configuration Write as Special Cycle request (Device = 1Fh, Function = 7h). If the target bus is the bridge's primary bus, the transaction is claimed and passed through as a Special Cycle. If the target bus is neither the primary bus nor in the range of buses defined by the bridge's secondary and subordinate bus registers, the transaction is claimed and passed through unchanged as a Type 1 Configuration Write. If the target bus is not the bridge's primary bus, but is within the range of buses defined by the bridge's secondary and subordinate bus registers, the transaction is ignored.

Table 16-2. Secondary Interface Supported Commands (Continued)

S_CBE[3:0]#	Command		Support
	Conventional PCI	PCI-X	
1100b	Memory Read Multiple	Split Completion	Conventional PCI—Treated as a Memory Read (S_CBE[3:0]#=0110b). PCI-X—Split Completion.
1101b	DAC	DAC	Lower 32 bits of the address driven out on S_AD[31:0], followed by the upper 32 bits.
1110b	Memory Read Line	Memory Read Block	Conventional PCI—Treated as a Memory Read (S_CBE[3:0]#=0110b). PCI-X—Memory Read Block.
1111b	Memory Write and Invalidate	Memory Write Block	Conventional PCI—Treated as a Memory Write (S_CBE[3:0]#=0111b). PCI-X—Memory Write Block.

## 17 BRIDGE BEHAVIOR

This section presents various scenarios of bridge behavior that occurs when the target responds to a cycle generated by the PCI 6540 on behalf of the initiating master.

### 17.1 BRIDGE ACTIONS FOR VARIOUS CYCLE TYPES

A PCI cycle is initiated by asserting FRAME#. In a bridge, there are several possibilities. Table 17-1 summarizes these possibilities, and delineates the PCI 6540 action for various cycle types.

After the PCI cycle is initiated, a target then has up to four cycles to respond before a Master Abort is initiated. If the target detects an address hit, it asserts

DEVSEL# in the cycle corresponding to the Configuration Status register DEVSEL# Timing bits (Transparent mode—PCISR[10:9]; PCI:06h or PCISSR[10:9], Non-Transparent mode—PCISR[10:9]; Primary PCI:06h, Secondary PCI:46h or PCISSR [10:9]; Primary PCI:46h, Secondary PCI:06h).

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

**Table 17-1. Bridge Actions for Various Cycle Types**

Initiator	Target	PCI 6540 Response
Master on Primary Port	Target on the same Primary Port	Does not respond. This situation is detected by decoding the address and monitoring P_DEVSEL# for other fast and medium devices on the primary port.
	Target on Secondary Port	Asserts P_DEVSEL# and terminates the cycle normally if posted; otherwise, returns with a Retry. Next, passes the cycle to the appropriate port. When cycle is complete on the target port, the PCI 6540 waits for the initiator to repeat the same cycle and end with normal termination.
	Target not on Primary nor Secondary Port	Does not respond and the cycle terminates as a Master Abort.
Master on Secondary Port	Target on the same Secondary Port	Does not respond.
	Target on Primary or other Secondary Port	Asserts S_DEVSEL# and terminates the cycle normally if posted; otherwise, returns with a Retry. Next, passes the cycle to the appropriate port. When cycle is complete on the target port, the PCI 6540 waits for the initiator to repeat the same cycle and end with normal termination.
	Target not on Primary nor other Secondary Port	Does not respond.

## 17.2 ABNORMAL TERMINATION (INITIATED BY BRIDGE MASTER)

### 17.2.1 Master Abort

A Master Abort indicates that the PCI 6540, operating as a master, receives no response from a target (*that is*, no target asserts P\_DEVSEL# or S\_DEVSEL#). The bridge de-asserts FRAME#, then de-asserts IRDY#.

## 17.3 PARITY AND ERROR REPORTING

Parity must be checked for all addresses and Write data. Parity is defined on the P\_PAR/P\_PAR64 and S\_PAR/S\_PAR64 signals. Parity should be even [*for example*, an even number of ones (1)] across AD, CBE#, and PAR. Parity information on PAR is valid the cycle after AD and CBE# are valid.

### 17.3.1 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, then placing two cycles into a high-impedance state after the bad address. P\_SERR# can be asserted only if the Command register P\_SERR# and Parity Error Response bits are both set to 1 (Transparent mode—PCICR[8, 6]=11b; PCI:04h, Non-Transparent mode—PCICR[8, 6]=11b; Primary PCI:04h, Secondary

PCI:44h). For Write Data phases, a Parity error is reported by asserting P\_PERR# two cycles after the Data phase and remains asserted for one cycle when PCICR[8]=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 causes the PCI bridge target to not claim the bus (P\_DEVSEL# remains inactive) and the cycle then terminates with a Master Abort. When the bridge is operating as master, a Data Parity error during a Read cycle results in the bridge master initiating a Master Abort.

## 17.4 S\_IDSEL MAPPING

When the PCI 6540 detects a Type 1 Configuration transaction for a device connected to the secondary port, it translates the Type 1 transaction-to-Type 0 transaction on the downstream interface. The Type 1 Configuration format uses a 5-bit field at P\_AD[15:11] as the Device Number, which the PCI 6540 translates to S\_AD[31:16]. Table 17-2 explains how the PCI 6540 generates S\_IDSEL. Devices are not allowed to connect IDSEL to AD16 (the source bridge is Device Number 0). The PCI 6540 is the source bridge for its secondary bus.

PCI-X systems drive the address *four* clocks before asserting FRAME# for Configuration transactions to allow additional setting time for the IDSEL input of devices.

Table 17-2. S\_IDSEL Generation

P_AD[15:11]	S_AD[31:16]	Device Number	S_AD Bit
00000b	0000_0000_0000_0001b	0 (Source Bridge)	16
00001b	0000_0000_0000_0010b	1	17
00010b	0000_0000_0000_0100b	2	18
...	...	...	...
01011b	0000_1000_0000_0000b	11	27
01100b	0001_0000_0000_0000b	12	28
...	...	...	...
01111b	1000_0000_0000_0000b	15	31
1xxxxb	0000_0000_0000_0000b	Special Cycle	N/A



## 17.5 32- TO 64-BIT CYCLE CONVERSION

When a 32-bit device generates a request to a 64-bit PCI target, the PCI 6540 can optionally convert this cycle to a 64-bit cycle on the target bus. The conversion is used only on 32-bit Prefetchable Read Memory and Posted Memory Write cycles with more than two Data transfers. This function is controlled through the Miscellaneous Options register Force 64-Bit Control bits (MSCOPT[15, 11]; PCI:46h).

If either Force 64-bit Control bit is set, all Posted Memory Write and Prefetchable Memory Read cycles are internally stored in the PCI 6540 as 64-bit cycles if the Data transfer is greater than 2 DWORD cycles. These cycles execute on the target following standard 64-bit PCI protocol.

The PCI 6540 asserts REQ64#, and if the target responds with ACK64# active, the PCI 6540 generates a 64-bit cycle. For Memory Write cycles, if the initial DWORD address is on an odd boundary, the PCI 6540 generates a 64-bit cycle with a value of Fh for the low DWORD of the initial Write Data transfer. If the target of a Posted Memory Write is a 32-bit device, and the target Retries with a Data transfer on an odd DWORD boundary, the remainder of the cycle is completed later as a 32-bit cycle when the PCI 6540 Retries it. Otherwise, the PCI 6540 continues to Retry the cycle as a 64-bit transaction.



## 18 PCI FLOW-THROUGH OPTIMIZATION

This section describes Flow-Through optimization, including precautions when using non-optimized PCI master devices, Posted Write and Delayed Read Flow Through, Read cycle optimization, and Read Prefetch boundaries.

### 18.1 OVERVIEW

The PCI 6540 operates in Flow-Through mode when data from the same transaction is simultaneously transferred on both sides of the bridge (*that is*, data on one side of the bridge “flows through” to the other side of the bridge). The PCI 6540 has several options to optimize PCI transfers after Flow-Through mode is achieved by way of the bridge.

The purpose of Flow-Through mode is to improve PCI Bus utilization and efficiency. If Data transfers on one side of the bridge are broken into several transactions on the other side of the bridge, poor bus efficiency results. By using Flow-Through mode, the PCI 6540 improves bus efficiency for Posted writes, Delayed Reads, and reads to prefetchable spaces.

### 18.2 PRECAUTIONS WHEN USING NON-OPTIMIZED PCI MASTER DEVICES

The PCI 6540 is capable of high-performance prefetching. However, some PCI masters may be unable to prefetch a large amount of data. This may be due to a small internal buffer size or other limiting factors. *For example*, if data is being read from a

register or FIFO-based architecture, valuable data may be lost if the host prematurely terminates a Prefetch cycle (ideally such spaces would not be listed as prefetchable). Under these circumstances the default values for prefetching may be overly aggressive and affect overall performance. In this case, tune default prefetching by reprogramming the Prefetch registers, as listed in Table 18-1. (Refer to Section 6, “Registers,” for a detailed description of these registers.)

The serial EEPROM can also be used to program the Configuration space upon reset.

### 18.3 POSTED WRITE FLOW THROUGH

During Flow Through of Posted Write cycles, if there is only one Data transfer pending in the Internal Post Memory Write queue, the PCI 6540 can be programmed to wait for a specified number of clocks before Disconnecting. The PCI 6540 de-asserts IRDY# on the target side and waits up to seven clocks for additional data from the initiator. If new Write data is received from the initiator during this period, the PCI 6540 re-asserts IRDY# and continues with the Write cycle. If new Write data is not received during this period, the PCI 6540 terminates the cycle to the target with the last data from the queue and later finishes the cycle.

The Flow-Through Control registers for Posted writes are detailed in Section 6, “Registers.” (Refer to PFTCR[2:0]; PCI:44h and SFTCR[2:0]; PCI:4Eh.)

**Table 18-1. Reprogramming Prefetch Registers**

Configuration Space Register	Data
Primary Initial Prefetch Count (PITLPCNT; PCI:48h)	Same value as Cache Line Size register (Transparent mode—PCICLSR; PCI:0Ch, Non-Transparent mode—PCICLSR; Primary PCI:0Ch, Secondary PCI:4Ch or PCISCLSR; Primary PCI:4Ch, Secondary PCI:0Ch). <b>Note:</b> Most PCs set this value to 08h.
Secondary Initial Prefetch Count (SITLPCNT; PCI:49h)	
Primary Incremental Prefetch Count (PINPCNT; PCI:4Ah)	0h
Secondary Incremental Prefetch Count (SINPCNT; PCI:4Bh)	0h
Primary Maximum Prefetch Count (PMAXPCNT; PCI:4Ch)	0h
Secondary Maximum Prefetch Count (SMAXPCNT; PCI:4Dh)	0h

## 18.4 DELAYED READ FLOW THROUGH

For Flow Through of Delayed Read cycles, if the Internal Read Queue is almost full, the PCI 6540 can be programmed to insert wait states to delay Read data from the target for a specified number of clocks before Disconnecting. During this time, the PCI 6540 de-asserts IRDY# on the target bus and waits up to seven clocks. If additional space becomes available in the Internal Read queue before the end of the IRDY# inactive period, the PCI 6540 re-asserts IRDY# and proceeds with the next Read Data phase. If no additional space becomes available in the Internal Read queue, the current Data phase becomes the last (IRDY# is asserted) and the cycle Disconnects at the end of the Data phase.

The Flow-Through Control registers for Delayed Reads are detailed in Section 6, "Registers." (Refer to PFTCR[6:4]; PCI:44h and SFTCR[6:4]; PCI:4Eh.)

## 18.5 READ CYCLE OPTIMIZATION

The main function of Read Cycle optimization is to increase the probability of Flow Through occurring during Read accesses to Prefetchable Memory regions. To improve the probability of Flow Through, the amount of data to be prefetched must be correctly configured.

If the PCI 6540 prefetches insufficient data, Flow Through does not occur because prefetching on the target side completes before the Initiator Retries the Read access. Under these circumstances, the Read cycles become divided into multiple cycles.

If the PCI 6540 prefetches excessive data and the internal FIFOs fill, the PCI 6540 must wait for the initiator to Retry the previous Read cycle and then flush the unclaimed data before queuing subsequent cycles.

The initial count is normally equivalent to the cache-line size. This assumes that a master usually requires at least one cache line of data. The incremental count is used only when the PCI 6540 does not detect Flow Through for the current cycle being prefetched during the Initial Prefetch Count. The PCI 6540 continues prefetching in increments until it reaches the maximum count, then Disconnects the cycle.

For Read prefetching, the PCI 6540 implements several registers that control the amount of data prefetched on the primary and secondary PCI Buses. The Prefetch registers listed in Table 18-1 can be used to optimize PCI 6540 performance during Read cycles.

The PCI 6540 prefetches until Flow Through occurs or prefetching must stop, based on the following conditions:

Prefetch continues while:

$$(IPMC + IPC + IPC + \dots + IPC) < MPC$$

where:

IPMC = Initial Prefetch Maximum Count

IPC = Incremental Prefetch Count,  $< \frac{1}{2} MPC$

MPC = Maximum Prefetch Count

If the Prefetch Count did not reach MPC and Flow Through was achieved, the PCI 6540 continues prefetching until the requesting master terminates the Prefetch request. Otherwise, when MPC is reached, the PCI 6540 stops prefetching data.

Incremental Prefetch can be disabled by setting  $IPC \geq MPC$ .

### 18.5.1 Primary and Secondary Initial Prefetch Count

Assuming that there is sufficient space in the internal FIFO, the Primary and Secondary Initial Prefetch Count registers (PITLPCNT; PCI:48h and SITLPCNT; PCI:49h, respectively) control the amount of data initially prefetched by the PCI 6540 on the primary or secondary bus during reads to the Prefetchable Memory region. If Flow Through is achieved during this initial prefetch, the PCI 6540 continues prefetching beyond this count.

### 18.5.2 Primary and Secondary Incremental Prefetch Count

The Primary and Secondary Incremental Prefetch Count registers (PINCPCNT; PCI:4Ah and SINCPNT; PCI:4Bh, respectively) control the amount of prefetching after the initial prefetch. If Flow Through is not achieved during the initial prefetch, the PCI 6540 attempts to prefetch further data, until the FIFO fills, or until the Maximum Prefetch Count is reached. Each subsequent prefetch is equal to the Incremental Prefetch Count.

### 18.5.3 Primary and Secondary Maximum Prefetch Count

The Primary and Secondary Maximum Prefetch Count registers (PMAXPNT; PCI:4Ch and SMAXPNT; PCI:4Dh, respectively) limit the amount of prefetched data for a single entry available in the internal FIFO at any time. During Read Prefetch cycles, the PCI 6540 Disconnects the cycle if the data count in the FIFO for the current cycle reaches this value, and Flow Through has not been achieved.

## 18.6 READ PREFETCH BOUNDARIES

For Memory Read and Memory Read Line commands, the PCI 6540 prefetches from the starting address up to an address with an offset that is a multiple of the Initial Prefetch Count. *For example*, if the starting address is 10h and the Initial Prefetch Count equals 20h, the PCI 6540 prefetches only a 10h (20h to 10h) count. After this, the PCI 6540 begins incremental prefetch until the Maximum Prefetch Count is reached, or Flow Through is achieved. The exception to this is in the case of a 64-bit request and six or fewer Dwords from the boundary, or a 32-bit request and four or fewer Dwords from the boundary, in which the PCI 6540 does not activate Incremental Prefetch.

For Memory Read Multiple commands, if the starting address is not 0, the PCI 6540 first prefetches from the starting address up to the address with an offset equal to that of the Initial Prefetch Count. After this, the PCI 6540 prefetches one additional Initial Prefetch Count. *For example*, if the starting address is 10h and the Initial Prefetch Count equals 20h, the PCI 6540 first prefetches a 10h (20h to 10h) count, then continues to prefetch another 20h count. Subsequent to this, Incremental Prefetch is invoked until the Maximum Prefetch Count is reached or Flow Through is achieved.



## 19 FIFO ARCHITECTURE

This section describes FIFO architecture, including how the FIFOs function with Memory Write and Read commands, and how to split the Read FIFO into four 1-KB blocks.

### 19.1 OVERVIEW

The PCI 6540 contains a 1-KB Write FIFO and 4-KB Read FIFO in both the downstream and upstream directions, for a total of 10 KB of data FIFO. The FIFO

architecture is designed for optimal PCI-X-to-PCI and PCI-X-to-PCI-X bridging, with the following features:

- Flow-Through Capable
- Programmable Prefetch Byte Counts of up to 2 KB on the PCI-X port
- Programmable Timeout Flushing or Command End Flush of Prefetched Data for PCI devices
- Segmentable into 1-KB FIFOs, dedicated to each of the four entries
- Controllable number of outstanding ADQs on the PCI-X port

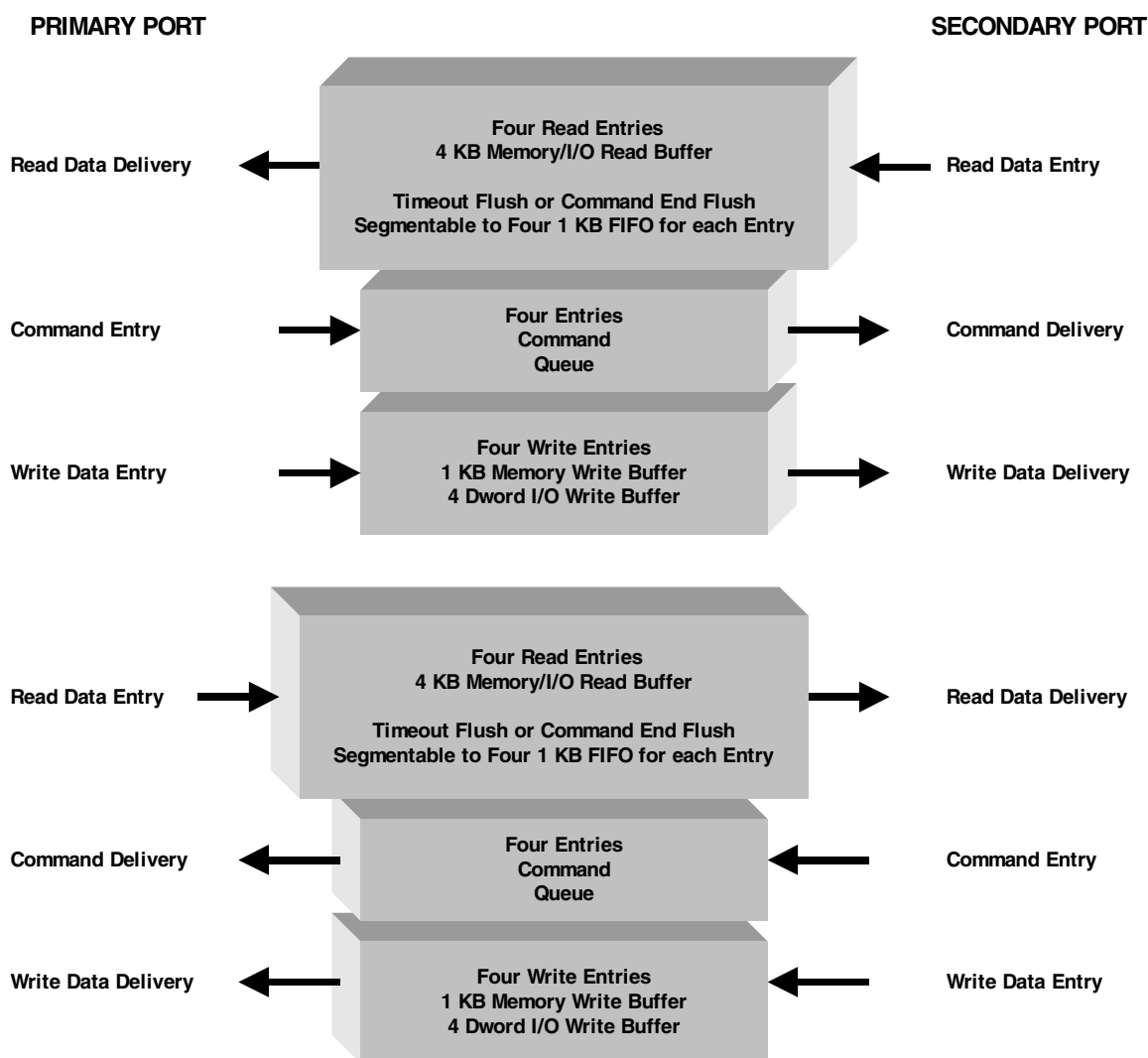


Figure 19-1. FIFO Architecture

## 19.2 MEMORY WRITES

This subsection describes PCI-to-PCI-X, PCI-X-to-PCI, and PCI-X-to-PCI-X Memory Writes.

### 19.2.1 PCI-to-PCI-X Memory Writes

If the initiator writes more than 64 bytes, the PCI 6540 absorbs the greatest amount of data possible into the Write FIFO. When the PCI-X port becomes available and there are 64 or more bytes available in the FIFO, the PCI 6540 begins data delivery from the FIFO to the PCI-X port, using the byte count present in the FIFO.

If the initiator Byte Count is less than 64 bytes, the PCI 6540 first receives all the bytes into the FIFO, then delivers the bytes to the PCI-X port.

### 19.2.2 PCI-X-to-PCI Memory Writes

When functioning as a target and receiving data into the Write FIFO, the PCI 6540 begins data delivery from the FIFO when the opposite port is available. The PCI-X byte count does not influence the start of data delivery to the PCI port.

### 19.2.3 PCI-X-to-PCI-X Memory Writes

When the PCI 6540 is a target, it always accepts the greatest amount of data possible into the Write FIFO. Concurrent with this process, when the opposite port is available and the incoming data reaches an ADB boundary, the PCI 6540 begins data delivery from the FIFO. If the initiator Byte Count is less than 128 bytes and has not reached an ADB boundary, the PCI 6540 first receives, then delivers, all the bytes.

## 19.3 MEMORY READS

This subsection describes PCI-to-PCI-X, PCI-X-to-PCI, and PCI-X-to-PCI-X Memory Reads.

### 19.3.1 PCI-to-PCI-X Memory Reads

When the PCI 6540 receives a PCI Read command, the PCI 6540 issues a Read command to the PCI-X port, using the programmed Prefetch Count specified by the following registers:

- Primary Initial Prefetch Count (PITLPCNT[2:1]; PCI:48h)
- Primary Incremental Prefetch Count (PINPCNT; PCI:4Ah)
- Secondary Initial Prefetch Count (SITLPCNT[2:1]; PCI:49h)
- Secondary Incremental Prefetch Count (SINPCNT; PCI:4Bh)

Regardless of the programmed Prefetch Count, the PCI 6540 does not prefetch beyond 16 cache lines. If the count is equal to, or larger than, the maximum programmed Outstanding ADQs allowed (default is 32 ADQs) in the PCI-X Downstream and Upstream Split Transaction registers (PCIXDNSTR; PCI:FCh and PCIXUPSTR; PCI:F8h, respectively), at least one PCI-X Read command is issued. In general, set the maximum programmed Outstanding ADQs to a value higher than the Prefetch Count so that the PCI 6540 can issue multiple PCI-X Read requests.

When data becomes available in the FIFO, the PCI 6540 begins data delivery to the PCI port.

Prefetched data in the PCI 6540 Read FIFO can either be flushed (if the PCI initiator finishes its current Read transaction) or preserved for a programmed time period. Upon timeout, if the PCI master has not returned to acquire additional data, the FIFO flushes the remaining data. This feature can greatly enhance the PCI-X Bus bandwidth, as the PCI 6540 can prefetch up to 16 cache lines of anticipated data for the PCI devices.



### 19.3.1.1 Prefetched Data Timeout Flushing

Prefetched data timeout flushing can be controlled by way of the register bits listed in Table 19-1.

**Table 19-1. Prefetched Data Timeout Flushing**

Control	Description
BUFCR[1]; PCI:4Fh	<b>Buffer Control Smart Prefetch Enable (Transparent Mode Only).</b> Amount of data prefetched is defined in the Maximum Prefetch Count registers (PMAXPCNT; PCI:4Ch and SMAXPCNT; PCI:4Dh). Defaults to 0. Values after a prefetch command: 0 = Remaining prefetched data is discarded upon completion of the current Read Command. 1 = Remaining prefetched data is <i>not</i> discarded, but remains available for the next Read Command with consecutive address. The prefetched data is only discarded upon a timeout. The timeout period can be programmed using the Smart Prefetch Timeout bits (BUFCR[6:5]; PCI:4Fh).
BCNTRL[8]; PCI:3Eh (Transparent mode) or BCNTRL[8]; PCI:42h (Non-Transparent mode)	<b>Bridge Control Primary Master Timeout.</b> Sets the maximum number of PCI clocks for an initiator on the primary bus to repeat the Delayed Transaction request. Values: 0 = Timeout after $2^{15}$ PCI clocks 1 = Timeout after $2^{10}$ PCI clocks Reset to 0.
BCNTRL[9]; PCI:3Eh (Transparent mode) or BCNTRL[9]; PCI:42h (Non-Transparent mode)	<b>Bridge Control Secondary Master Timeout.</b> Sets the maximum number of PCI clocks for an initiator on the secondary bus to repeat the Delayed Transaction request. Values: 0 = Timeout after $2^{15}$ PCI clocks 1 = Timeout after $2^{10}$ PCI clocks Reset to 0.

### 19.3.1.2 Setting the Prefetch Count

#### 19.3.1.2.1 PCI Read from PCI-X Port

For PCI master reads from PCI-X devices, the PCI 6540 can be set up to prefetch data from the PCI-X port to enhance system performance by minimizing PCI access to the PCI-X Bus.

When the PCI 6540 receives a PCI Read command, the PCI 6540 issues a Read command to the PCI-X port, using the programmed Prefetch Count specified by the following registers:

- Primary Initial Prefetch Count (PITLPCNT[2:1]; PCI:48h)
- Primary Incremental Prefetch Count (PINPCNT; PCI:4Ah)
- Secondary Initial Prefetch Count (SITLPCNT[2:1]; PCI:49h)
- Secondary Incremental Prefetch Count (SINPCNT; PCI:4Bh)

#### 19.3.1.2.2 PCI Read from PCI Port

For details on PCI Read from PCI port, refer to Section 18, “PCI Flow-Through Optimization.”

### 19.3.2 PCI-X-to-PCI Memory Reads

After receiving the PCI-X Memory Read command, the PCI 6540 issues a PCI Memory Read command to the PCI port when the port becomes available. When sufficient data is received to read an ADB boundary, the PCI 6540 begins forwarding the Read data to the PCI-X port.

If there is a mid-transaction wait period during which the PCI port is not supplying the PCI 6540 with sufficient data to reach the next ADB boundary, the PCI-X Split Completion cycle is Disconnected so that the PCI 6540 can serve another PCI-X master.

### 19.3.3 PCI-X-to-PCI-X Memory Reads

Depending on the programmed Outstanding ADQs Limit (default is 32 ADQs) in the PCI-X Downstream and Upstream Split Transaction registers (PCIXDNSTR; PCI:FCh and PCIXUPSTR; PCI:F8h, respectively)—the PCI 6540 issues a Read command to the opposite port until the requested and pending Read Byte Counts have reached the maximum programmed Outstanding ADQs Limit.

## 19.4 SPLITTING THE READ FIFO INTO FOUR 1-KB BLOCKS—TRANSPARENT MODE

Normally, the PCI 6540 dynamically allocates FIFO areas for each entry. By setting the Buffer Control Split FIFO Enable bit (BUFCR[2]=1; PCI:4Fh), the designer can dedicate 1 KB of FIFO for each entry. This allows special applications (*such as* DSP processors) to prefetch and store a fixed amount of data in the PCI 6540 while awaiting processing.

If the PCI master requesting the data is unable to transfer the 1 KB stored in the FIFO within one transaction, the PCI 6540 Timeout Flushing mechanism can be used (refer to Section 19.3.2) to preserve the data in the FIFO for a programmed period of time. This allows the PCI master to transfer the data in several segments. This feature may also be used to overcome latency requirements or restrictions that may apply to this bus segment.

## 20 NON-TRANSPARENT MODE

This section provides an overview of Non-Transparent mode, and describes XB\_MEM input use to avoid initial Retry latency, interrupts, and the mode power-up sequence.

### 20.1 OVERVIEW

The PCI 6540 Non-Transparent mode allows bridging between two independent processor domains, with the host processor connected to the PCI 6540 primary port. A second processor can reside on the secondary port and perform initialization, control, and I/O functions within its own domain, without interference from the primary bus host. The subsystem connected to the secondary port is not visible to the primary port host processor, and driver software operating on the host bus recognizes the presence of the Non-Transparent bridge and correctly manages the resources.

Moreover, the PCI 6540 also performs address translation between the two PCI ports, allowing for communication between the two domains. The PCI 6540 provides three base address registers (BARs) on each side of the bridge to specify which cycles are passed downstream or upstream, after being translated using the values in the Address Translation Control registers. Non-Transparent mode is enabled through the TRANS# pin (TRANS#=1).

The PCI 6540 provides the following Non-Transparent capabilities:

- Downstream and upstream address translation
- Separate Configuration space for primary and secondary interfaces
- Up to three separate address ranges that can be specified with standard BAR definitions
- 32-bit I/O, 32-bit Memory, and 64-bit Memory Address Translation support
- Serial EEPROM-loadable Address Translation registers (allowing Non-Transparent operation without additional software control) (refer to Section 6.2.4.13)
- Powerful Message register mechanism, doorbells, status, and events with interrupt capability to pass information from one side of the bridge to the other

If a programmed serial EEPROM is connected to the PCI 6540, the serial EEPROM contents begin loading into the PCI 6540 registers upon reset removal. During this loading process, any Configuration cycle from either PCI port results in a Retry response. Depending on which port is set to a higher boot priority by the P\_BOOT input, the lower priority boot master's accesses to the PCI standard BARs are Retried, unless XB\_MEM=1. (Refer to Section 20.2 for further details.) Accesses to other Configuration registers are not affected. Upon RSTIN# assertion, the PORT\_READY Status bit is cleared (DWNTNE[31]=0; Ext:0Fh or UPSTNE[31]=0; Ext:0Bh, respectively) to indicate that the port is not ready for access by the controlling processor.

The higher priority boot master (in general, this is the secondary port intelligent subsystem) allocates a Memory and/or I/O region that can be accessed by the lower boot priority host (in general, the primary port host). At that time, the higher priority boot master sets the P\_PORT\_READY or S\_PORT\_READY bit. After this bit is set, the Retried BAR Access Configuration cycle from the lower priority boot master can proceed, and the lower priority boot master can proceed with normal PCI initialization to set up the desired Memory/I/O space allocation.

The PCI 6540 architecture provides semaphore mechanisms that can be used to ensure exclusive access to shared registers. There are also multiple cross-bridge interrupt mechanisms available. A direct interrupt mechanism allows user-encoded messages to be written to registers that can cause interrupts. Designers must decide on the definitions used in the Message registers. There are also 16 Doorbell registers for cross-bridge communication. Port reset or power-down can also be configured to cause interrupts to the opposite port.

## 20.2 USING XB\_MEM INPUT TO AVOID INITIAL RETRY LATENCY

The P\_PORT\_READY or S\_PORT\_READY mechanism (which results in a Retry for BAR Access Configuration cycles if the subsystem is not set up) is disabled if the XB\_MEM input pin is set to high. In this case, the PCI 6540 provides a hardcoded 16 MB Memory space Cross-Bridge Communication window at power-up. The PCI 6540 automatically claims this 16 MB of Memory space. This allows boot-up of the lower priority boot port to proceed, without waiting for the higher priority boot port to program the corresponding Memory BARs. When XB\_MEM=1 (PRV\_DEV pin in Transparent mode), the P\_PORT\_READY or S\_PORT\_READY mechanism is not relevant and BAR accesses are not Retried. Although the default claims 16 MB, the BARs can be modified by serial EEPROM or software to change the window size.

## 20.3 INTERRUPTS

Message Signaled Interrupts (MSI) are non-shared interrupts that enforce data consistency. The system guarantees that any data written by the device prior to sending the MSI has reached its final destination before the interrupt service routine accesses the data. MSI enables the PCI 6540 to request service by writing a system-specified message to a system-specified address (PCI DWORD Memory Write transaction). The Transaction address specifies the message destination and the Transaction data specifies the message. System software initializes the message destination and message during device configuration.

### 20.3.1 Direct Message Interrupts

The PCI 6540 has four upstream and four downstream Message registers which when written to, can generate immediate interrupts to the other side of the bridge. These are the fastest interrupt mechanisms that the PCI 6540 has, and are faster in latency than standard Doorbell interrupts for software applications.

When a PCI master needs to communicate with the host on the other side of the PCI 6540 bridge, the master can use any one of the four Message registers. When the master writes an encoded message into the Message register, the write generates an interrupt to the host. The interrupt service routine can first read the Interrupt Status registers to determine which Message Status bit is set and read the Message register to acquire the Interrupt message. The service routine should then write 1 to the corresponding Status bit to clear the bit. This allows the service routine to quickly react to the encoded message without polling several registers. (Refer to Section 6.2.4.8 for further details.)

### 20.3.2 Doorbell Interrupts

The PCI 6540 has 16 upstream and 16 downstream doorbell interrupt registers which when written to, can generate immediate interrupts to the other side of the bridge.

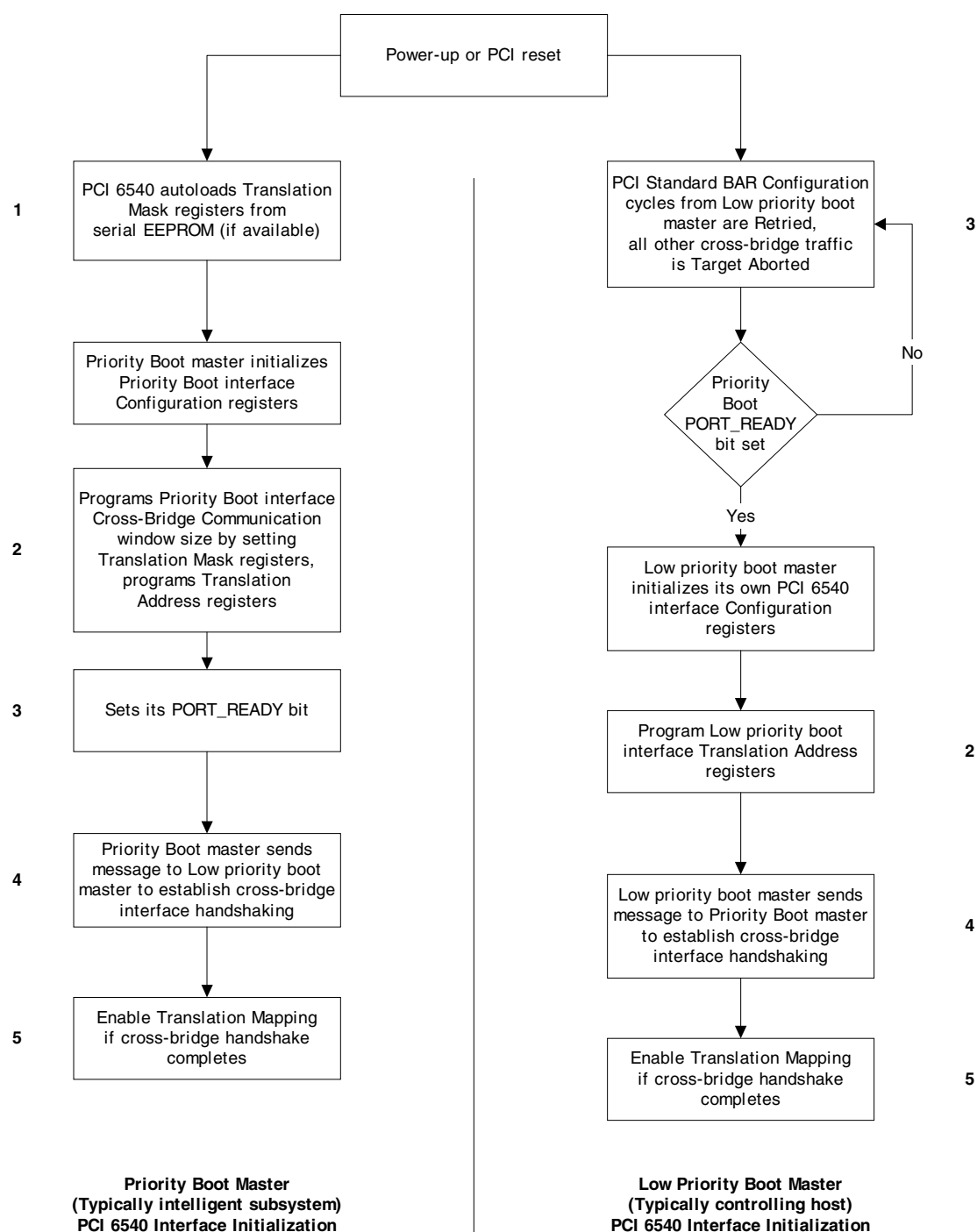
When a PCI master needs to communicate with the host on the other side of the PCI 6540 bridge, it can make use any of the 16 doorbell interrupts. When the requesting master first writes 1, then 0 to its Doorbell Interrupt Request bit, an interrupt is automatically generated to the host. The interrupt service routine can read the Doorbell Status register to find out which device is requesting the interrupt and can then inquire the corresponding device. The service routine should then write 1 to the corresponding Status bit to clear the bit. (Refer to Section 6.2.4.10 for further details.)

### 20.3.3 Message Signaled Interrupts

Refer to Section 6.2.4.9.

## 20.4 NON-TRANSPARENT MODE POWER-UP/PCI RESET SEQUENCE

Figure 20-1 delineates the Non-Transparent mode power-up/PCI reset sequence.

**Notes:**

1. Translation Mask register values are read from serial EEPROM offsets 32h to 33h and 3Eh to 3Fh.
2. Extended registers at indexed address Ch to Eh (primary port) and 8h to Ah (secondary port).
3. PORT\_READY bits, at Extended register indexed address Fh (primary port) and Bh (secondary port) are cleared upon P\_RSTIN# and S\_RSTIN# assertion.
4. Handshaking can be achieved using Direct Message Interrupts at registers A4h to ABh. Handshaking messages are user-defined status/command information.
5. Translation can be enabled at Extended register indexed address 0Fh (primary port) and 0Bh (secondary port).

**Figure 20-1. Non-Transparent Mode Power-Up/PCI Reset Sequence**



## 21 POWER MANAGEMENT

This section describes the Power Management feature and P\_PME# and S\_PME# use.

### 21.1 OVERVIEW

The PCI 6540 incorporates functionality that meets the requirements of *PCI Power Mgmt. r1.1*. These features include:

- PCI Power Management registers, using the Enhanced Capabilities Port (ECP) address mechanism
- Support for D<sub>0</sub>, D<sub>3hot</sub> and D<sub>3cold</sub> power management states
- Support for D<sub>0</sub>, D<sub>1</sub>, D<sub>2</sub>, D<sub>3hot</sub> and D<sub>3cold</sub> power management states for devices behind the bridge
- Support for B<sub>2</sub> secondary bus power state when in the D<sub>3hot</sub> power management state

Table 21-1 delineates the states and related actions the PCI 6540 performs during Power Management transitions. (No other transactions are allowed.)

**Table 21-1. States and Related Actions during Power Management Transitions**

Current State	Next State	Action
D <sub>0</sub>	D <sub>3cold</sub>	Power removed from the PCI 6540. A power-up reset must be performed to bring the PCI 6540 to D <sub>0</sub> .
D <sub>0</sub>	D <sub>3hot</sub>	If enabled by the BPCC_EN pin, the PCI 6540 disables the secondary clocks and drives them low.
D <sub>0</sub>	D <sub>2</sub>	During an unimplemented power state, the PCI 6540 ignores the write to the Power State bits (power state remains at D <sub>0</sub> ).
D <sub>0</sub>	D <sub>1</sub>	
D <sub>3hot</sub>	D <sub>0</sub>	The PCI 6540 enables secondary clock outputs and performs an internal chip reset. S_RSTOUT# is not asserted. All registers are returned to the reset values and buffers are cleared.
D <sub>3hot</sub>	D <sub>3cold</sub>	Power removed from the PCI 6540. A power-up reset must be performed to bring the PCI 6540 to D <sub>0</sub> .
D <sub>3cold</sub>	D <sub>0</sub>	During a power-up reset, the PCI 6540 performs the standard power-up reset functions.

### 21.2 P\_PME# AND S\_PME# SIGNALS

In Transparent mode, P\_PME# and S\_PME# are not used and should be tied high.

In Non-Transparent mode, depending on their setting, P\_BOOT, P\_PME# and S\_PME# are passed from the high-priority boot port to the low-priority boot port.





# 22 HOT SWAP

This section describes the Hot Swap feature and its use.

## 22.1 OVERVIEW

The PCI 6540 incorporates functionality that meets *PICMG 2.1 R2.0* requirements with High-Availability Programming Interface level 1 (PI=1). The CompactPCI Hot Swap register block is located at PCI Configuration offset E4h. Refer to *PICMG 2.1 R2.0* for detailed implementation guidelines. The Hot Insertion Power-Up sequence recommendation is illustrated in Figure 22-1.

**Note:** If the Hot Swap feature is not needed, EJECT input must be connected to logic 0. Otherwise, the PCI 6540 does not function. ENUM# and L\_STAT may remain unconnected.

## 22.2 EARLY POWER SUPPORT

The PCI 6540 incorporates Early Power Support in the following way:

- Tolerates backend interface not being powered when fully powered by Early Power. The PCI 6540 places all port PCI signals (except S\_ACK64#) into a high-impedance state until its corresponding RSTIN# signal is de-asserted.
- When fully powered by backend power, the PCI 6540 places all port PCI signals (except S\_ACK64#) into a high-impedance state until its corresponding RSTIN# signal is de-asserted. (Refer to Table 5-2, “Pin State during PWRGD, P\_RSTIN#, S\_RSTIN#, and Device Hiding,” on page 5-9 for further details.)

## 22.3 HOT SWAP SIGNALS

The PCI 6540 uses the following Hot Swap-related pins:

- **ENUM#**—Output signal used to notify the system host that a board was freshly inserted or is about to be extracted. ENUM# is an open drain signal. ENUM# is asserted if the INS or EXT bit is set and EIM is 0 (HS\_CSR[7, 1]=10b or HS\_CSR[6, 1]=10b; PCI:E6h, respectively).
- **L\_STAT**—Status BLUE LED. LED is ON if RSTIN# is asserted. LED is also ON when the LOO bit is set (HS\_CSR[3]=1; PCI:E6h) and RSTIN# is de-asserted. The LED is an active high signal that allows other circuits to drive the BLUE LED.
- **EJECT**—Handle Switching input. This signal should be debounced by external hardware and must be connected to logic 0, if the Hot Swap function is not used. This signal can cause ENUM# assertion.

The recommended GPIO pin (GPIO7) is also designated for Hot Swap use:

- **HEALTHY# (GPIO7)**—Used as the board HEALTHY# output. This pin has an internal weak pull-up resistor. Subsystem software can set the pin to output the desired Board Healthy status to the system and to control the custom logic-generated Hot Swap port RSTIN# signal.

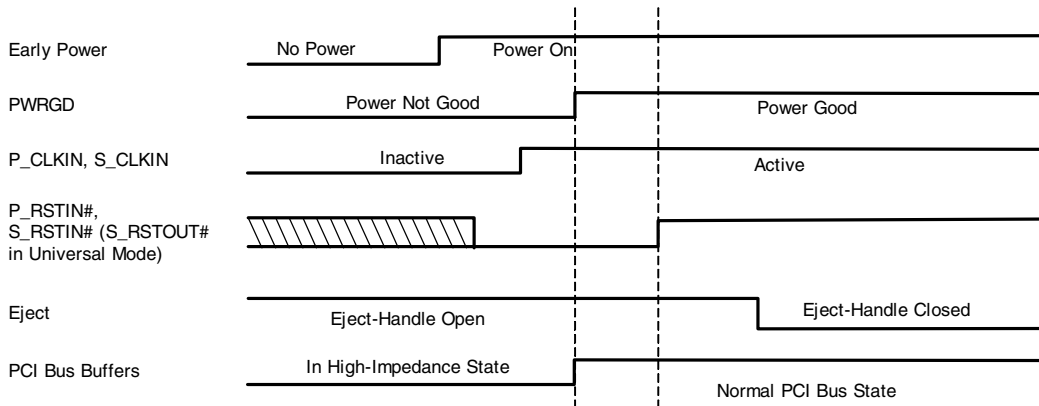


Figure 22-1. Hot Insertion Power-Up Sequence Recommendation

## 22.4 HOT SWAP REGISTER CONTROL AND STATUS

The PCI 6540 Hot Swap Control/Status register (HS\_CSR) is located at PCI offset E6h.

## 22.5 AVOIDING INITIALLY RETRY OR INITIALLY NOT RESPONDING REQUIREMENT

The P\_PORT\_READY or S\_PORT\_READY mechanism, which results in Retry for BAR Access Configuration cycles if the subsystem is not set up, may be disabled if the XB\_MEM input pin is set to high. In this case, the PCI 6540 utilizes a cross-bridge communication window default of 16 MB Memory space. The PCI 6540 automatically claims this 16 MB of Memory space, which allows boot-up of the low-priority boot port to proceed, without waiting for the priority boot port to program the corresponding Memory BARs. When XB\_MEM=1 (PRV\_DEV pin in Transparent mode), the P\_PORT\_READY or S\_PORT\_READY mechanism is not relevant and access to BARs is not Retried. Although the default claims 16 MB, the BARs can be altered by serial EEPROM or software to change the window size.

During reset, the PCI 6540 is a Not Responding device. Therefore, GPIO7 may be used, *for example*, to generate HEALTHY# control by the subsystem to control the LOCAL\_PCI\_RST# input to the PCI 6540 Hot Swap port.

## 22.6 DEVICE HIDING

The PCI 6540 implements Device Hiding to eliminate mid-transaction extractions. This invokes Device Hiding by hardware from the Hot Swap port after RSTIN# becomes inactive and the ejector handle remains unlocked.

Software quiesces the PCI 6540 when Device Hiding is invoked. The current transaction is completed as early as possible. The PCI 6540 does not initiate a transaction as a master, respond as a target to I/O transactions, nor signal interrupts.

When Device Hiding is invoked, the PCI 6540 terminates the current Configuration transaction by signaling a Disconnect. After the current transaction completes (is Disconnected), the PCI 6540 does not respond as a target to any subsequent transactions until Device Hiding is canceled.

If not participating in a transaction when Device Hiding is invoked, the PCI 6540 does not respond as a target to subsequent transactions until Device Hiding is canceled.

Device Hiding is canceled when the handle switch is relocked.

## 22.7 IMPLEMENTING HOT SWAP CONTROLLER USING PCI 6540 GPIO PINS

In Transparent mode, GPIO[15:8] can be used to connect to the radial BD\_SEL# signal. GPIO[7:0], which have weak internal pull-up resistors, can be used for the radial HEALTHY# signal. ENUM# can be used to trigger HEALTHY# inquiry by reading the GPIO ports.

## 22.8 RUNNING PCI-X IN HOT SWAP SLOTS

To implement the primary port to run on PCI-X protocol, set P\_XCAP input to 1, although the standard PCI-X reset initialization sequence is not available, as in Hot Swap case. To implement the secondary port to run on PCI-X protocol, set S\_XCAP\_IN to 1.

When no initialization pattern is issued during reset, pull P\_XCAP or S\_XCAP\_IN high to force the primary or secondary interface to run in PCI-X mode, or pull these signals low to run in PCI mode. In the latter case, do **not** connect S\_XCAP\_PU to S\_XCAP\_IN.

## 23 VPD

This section describes the VPD feature.

The PCI 6540 contains the Vital Product Data (VPD) registers, as specified in *PCI r2.3*. VPD information is stored in the serial EEPROM device, along with Autoload information.

The PCI 6540 provides storage of 192 bytes of VPD data in the serial EEPROM device.

The VPD register block is located at offsets E8h to ECh in PCI Configuration space. VPD also uses the Enhanced Capabilities Port Address mechanism.



# 24 TESTABILITY/DEBUG

This section describes the JTAG interface for use in testing and debugging the PCI 6540.

## 24.1 JTAG INTERFACE

The PCI 6540 provides a JTAG Boundary Scan interface which can be utilized to debug a pin's board connectivity.

### 24.1.1 IEEE 1149.1 Test Access Port

The IEEE 1149.1 Test Access Port (TAP), commonly called the JTAG (Joint Test Action Group) debug port, is an architectural standard described in IEEE Standard 1149.1-1990, *IEEE Standard Test Access Port and Boundary-Scan Architecture*. The standard describes a method for accessing internal chip facilities using a four- or five-signal interface.

The JTAG debug port, originally designed to support scan-based board testing, is enhanced to support the attachment of debug tools. The enhancements, which comply with IEEE Standard 1149.1-1990 specifications for vendor-specific extensions, are compatible with standard JTAG hardware for boundary-scan system testing.

- **JTAG Signals**—JTAG debug port implements the four required JTAG signals—TCK, TDI, TDO, TMS—and the optional TRST# signal. (Refer to Table 3-9 for signal descriptions.)
- **JTAG Clock Requirements**—The TCK signal frequency can range from DC to 10 MHz.
- **JTAG Reset Requirements**—JTAG debug port logic and system simultaneously reset. The two methods for placing the PCI 6540 JTAG TAP controller into the Test-Logic-Reset state are as follows:
  - Upon receiving TRST#, the JTAG TAP controller returns to the Test-Logic Reset state
  - Hold the PCI 6540 TMS pin high while transitioning the PCI 6540 TCK pin five times

### 24.1.2 JTAG Instructions

The JTAG debug port provides the standard **EXTEST**, **SAMPLE/PRELOAD**, **CLAMP**, **HIGHZ**, **IDCODE**, and **BYPASS** instructions. Invalid instructions behave as **BYPASS** instructions.

The PCI 6540 returns the IDCODE values listed in Table 24-1. Table 24-2 lists the JTAG instructions, along with their input codes.

Table 24-1. PCI 6540 JTAG IDCODE Value

M																													L			
S																													S			
B																													B			
3	3	2	2	2	2	2	2	2	2	2	1	1	1	1	1	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0	
1	0	9	8	7	6	5	4	3	2	1	0	9	8	7	6	5	4	3	2	1	0											
Version				Part Number (PCI 6540 when converted to decimal)																PLX Manufacturer Identity												
0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	1	0	1	1	0	0	0	1	0	1

Table 24-2. JTAG Instructions (IEEE Standard 1149.1-1990)

Instruction	Input Code	Instruction	Input Code
EXTEST	00000b	HIGHZ	00101b
SAMPLE/PRELOAD	00001b	IDCODE	00110b
CLAMP	00100b	BYPASS	11111b

### 24.1.3 JTAG Boundary Scan

Boundary Scan Description Language (BSDL), IEEE 1149.1b-1994, is a supplement to IEEE Standard 1149.1-1990 and IEEE 1149.1a-1993, *IEEE Standard Test Access Port and Boundary-Scan Architecture*. BSDL, a subset of the IEEE 1076-1993 Standard VHSIC Hardware Description Language (VHDL), allows a rigorous description of testability features in components which comply with the standard. Automated test pattern generation tools use BSDL for package interconnect tests and Electronic Design Automation (EDA) tools for synthesized test logic and verification. BSDL supports robust extensions that can be used for internal test generation and to write software for hardware debug and diagnostics.

The primary components of BSDL include the logical port description, physical pin map, instruction set, and Boundary register description.

The logical port description assigns symbolic names to the PCI 6540 pins. Each pin has a logical type of in, out, in out, buffer, or linkage that defines the logical direction of signal flow.

The physical pin map correlates the PCI 6540 logical ports to the physical pins of a specific package. A BSDL description can have several physical pin maps; each map is given a unique name.

Instruction set statements describe the bit patterns that must be shifted into the Instruction register to place the PCI 6540 in the various Test modes defined by the standard. Instruction set statements also support descriptions of instructions that are unique to the PCI 6540.

The Boundary register description lists each of its cells or shift stages. Each cell has a unique number—the cell numbered 0 is the closest to the Test Data Out (TDO) pin and the cell with the highest number is closest to the Test Data In (TDI) pin. Each cell contains additional information, including:

- Cell type
- Logical port associated with the cell
- Logical function of the cell
- Safe value
- Control cell number
- Disable value
- Result value

### 24.1.4 JTAG Reset Input TRST#

The TRST# input pin is the asynchronous JTAG logic reset. TRST# assertion causes the PCI 6540 TAP controller to initialize. In addition, when the TAP controller is initialized, it selects the PCI 6540 normal logic path (core-to-I/O). Take the following into consideration when implementing the asynchronous JTAG logic reset on a board:

- If JTAG functionality is required, one of the following should be considered:
  - Use the TRST# input signal low-to-high transition once.
  - Hold the PCI 6540 TMS pin high while transitioning the PCI 6540 TCK pin five times.
- If JTAG functionality is not required, the TRST# signal must be directly connected to ground.

**Note:** IEEE Standard 1149.1-1990 requires pull-up resistors on the TDI, TMS, and TRST# pins. To remain PCI r2.3-compliant, no internal pull-up resistors are provided on JTAG pins in the PCI 6540; therefore, the pull-up resistors must be externally added to the PCI 6540 when implementing JTAG.

## 25 ELECTRICAL SPECS

This section presents the PCI 6540 electrical specifications.

### 25.1 GENERAL ELECTRICAL SPECIFICATIONS

The ratings provided in this subsection are those above which the useful life of the PCI 6540 may be impaired.

Use heatsinks when operating in a 133 MHz environment. Table 25-1 lists the PCI 6540 maximum ratings. Table 25-2 lists the PCI 6540 functional operating range. Table 25-3 lists the PCI 6540 DC electrical characteristics.

**Caution:** *Stresses greater than the maximums listed in Table 25-1 cause permanent damage to the PCI 6540. This is a stress rating only and functional operation of the PCI 6540 at or above those indicated in the operational sections of this data book is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.*

**Table 25-1. Maximum Ratings**

Parameter	Minimum	Maximum	Parameter	Minimum	Maximum
Storage Temperature Range	-55 °C	+125 °C	Maximum Voltage to Signal Pins	—	5.5V
Junction Temperature	—	+125 °C	Maximum Power	—	3.0W
V <sub>DD_IO</sub> Supply Voltage	—	3.9V	Maximum V <sub>DD_IO</sub> Power (output load dependent)	—	1.2W
V <sub>DD_CORE</sub> Supply Voltage	—	3.0V	Maximum V <sub>DD_CORE</sub> Power	—	1.8W
Analog V <sub>DD</sub> Supply Voltage	—	3.0V	Maximum Analog V <sub>DD</sub> Power	—	50 mW

**Table 25-2. Functional Operating Range**

Parameter	Minimum	Maximum	Parameter	Minimum	Maximum
V <sub>DD_IO</sub> Supply Voltage	3.0V	3.6V	Analog V <sub>DD</sub> Supply Voltage	2.25V	2.75V
V <sub>DD_CORE</sub> Supply Voltage	2.25V	2.75V	Operating Ambient Temperature	0 °C	70 °C

Table 25-3. DC Electrical Characteristics

Symbol	Parameter	Condition	Minimum	Maximum	Unit	Notes
$V_{DD\_IO}$	$V_{DD\_IO}$ Supply Voltage	—	3.0	3.6	V	
$V_{DD\_CORE}$ $P/S\_A_{VDD}$	$V_{DD\_CORE}$ , $P/S\_A_{VDD}$	—	2.25	2.75	V	
$V_{ih}$	Input High Voltage	—	$0.5 V_{DD\_IO}$	$V_{DD\_IO}$	V	
$V_{il}$	Input Low Voltage	—	-0.5	$+0.3 V_{DD\_IO}$	V	
$V_{ol}$	Output Low Voltage	$I_{iout} = +1500 \mu A$	—	$+0.1 V_{DD\_IO}$	V	
$V_{oh}$	Output High Voltage	$I_{iout} = -500 \mu A$	$0.9 V_{DD\_IO}$	—	V	
$I_{il}$	Input Leakage Current	$0 < V_{in} < V_{DD\_IO}$	—	$\pm 2$	$\mu A$	
$C_{in}$	Input Pin Capacitance	—	—	7.0	pF	



## 25.2 PLL AND CLOCK JITTER

The PCI 6540 uses one PLL for the primary interface and one PLL for the secondary interface. These PLLs can be individually disabled by connecting their corresponding P\_PLEN# or S\_PLEN# pin to 1. During slow speed applications for a port where PLL should be disabled (for example, at 33 MHz), set the corresponding P\_PLEN# or S\_PLEN# input to high.

The PLL is sensitive to power and ground noise. Dedicated and nearby PLL power and ground pins help prevent malfunctions caused by potentially large power and/or ground bounce. To eliminate this problem, connect the PLL  $V_{DD}$  pins to a clean 2.5V supply.

**Table 25-4. PLL and Clock Jitter Parameters**

Parameter	Minimum	Typical	Maximum	Unit	Condition
Input Frequency	50	—	133	MHz	—
Input Rise and Fall Time	—	—	500	ps	—
Input Cycle-to-Cycle Jitter	-100	—	+100	ps	—
Input Jitter Modulation Frequency	Must be < 100 KHz to allow PLL tracking or > 30 MHz to allow PLL filtering				
Output Cycle-to-Cycle Jitter	-150	—	+150	ps	Clean Power $V_{DD} = 2.5V$
Output Duty Cycle	45	—	55	%	Clean Power $V_{DD} = 2.5V$
Phase Lock Time	—	—	100	$\mu s$	Clean Power $V_{DD} = 2.5V$
PLL Power Dissipation	—	9	25	mW	Clean Power $V_{DD} = 2.5V$ $F_{in} = F_{out} = 133 \text{ MHz}$
Operating Temperature	-40	—	+125	$^{\circ}C$	—

## 25.3 PCI-X/PCI SIGNAL TIMING SPECIFICATION

Figure 25-1 illustrates the PCI 6540 signal timing specifications. Table 25-5 delineates the minimum and maximum values, for 133 MHz PCI-X and 66 MHz PCI, for the symbols that appear in Figure 25-1.

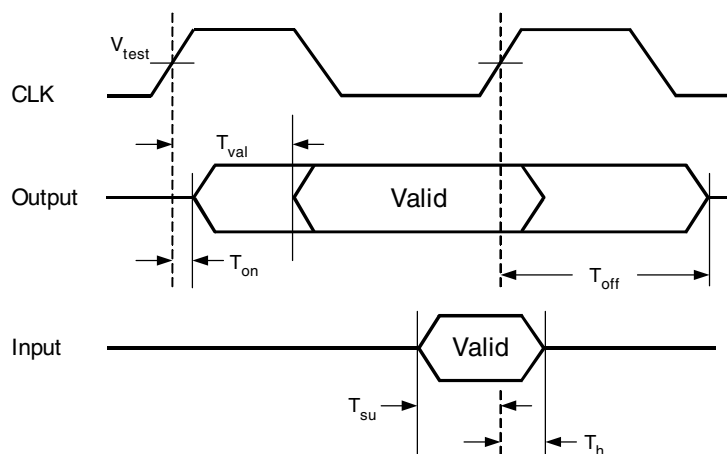


Figure 25-1. PCI-X/PCI Signal Timing Specification

**Note:** Refer to PCI-X r1.0b for detailed descriptions of the symbols that appear in Figure 25-1.

Table 25-5. 133 MHz PCI-X and 66 MHz PCI Signal Timing for Figure 25-1

Symbol	Parameter	133 MHz PCI-X		66 MHz PCI		Unit
		Minimum	Maximum	Minimum	Maximum	
$T_{val}$	CLK to Signal Valid Delay—Bused Signals	0.7	3.8	2	6	ns
$T_{val(ftp)}$	CLK to Signal Valid Delay—Point to Point	0.7	3.8	2	6	ns
$T_{on}$	Float to Active Delay	0	—	2	—	ns
$T_{off}$	Active to Float Delay	—	7	—	14	ns
$T_{su}$	Input Setup Time to CLK—bused signals	1.2	—	3	—	ns
$T_{su(ftp)}$	Input Setup Time to CLK—Point to Point	1.2	—	5	—	ns
$T_h$	Input Signal Hold Time from CLK	0.5	—	0	—	ns
$V_{test}$	Voltage Test	—	0.4	—	0.4	$V_{DD}$

# 26 MECHANICAL SPECS

This section provides the PCI 6540 mechanical dimensions and pinout.

## 26.1 MECHANICAL DIMENSIONS

The PCI 6540 uses an industry standard 27 x 27 mm 380-pin (ball) PBGA.

Figure 26-1 illustrates the mechanical dimensions. Table 26-1 lists the mechanical dimensions, in millimeters, unless specified otherwise.

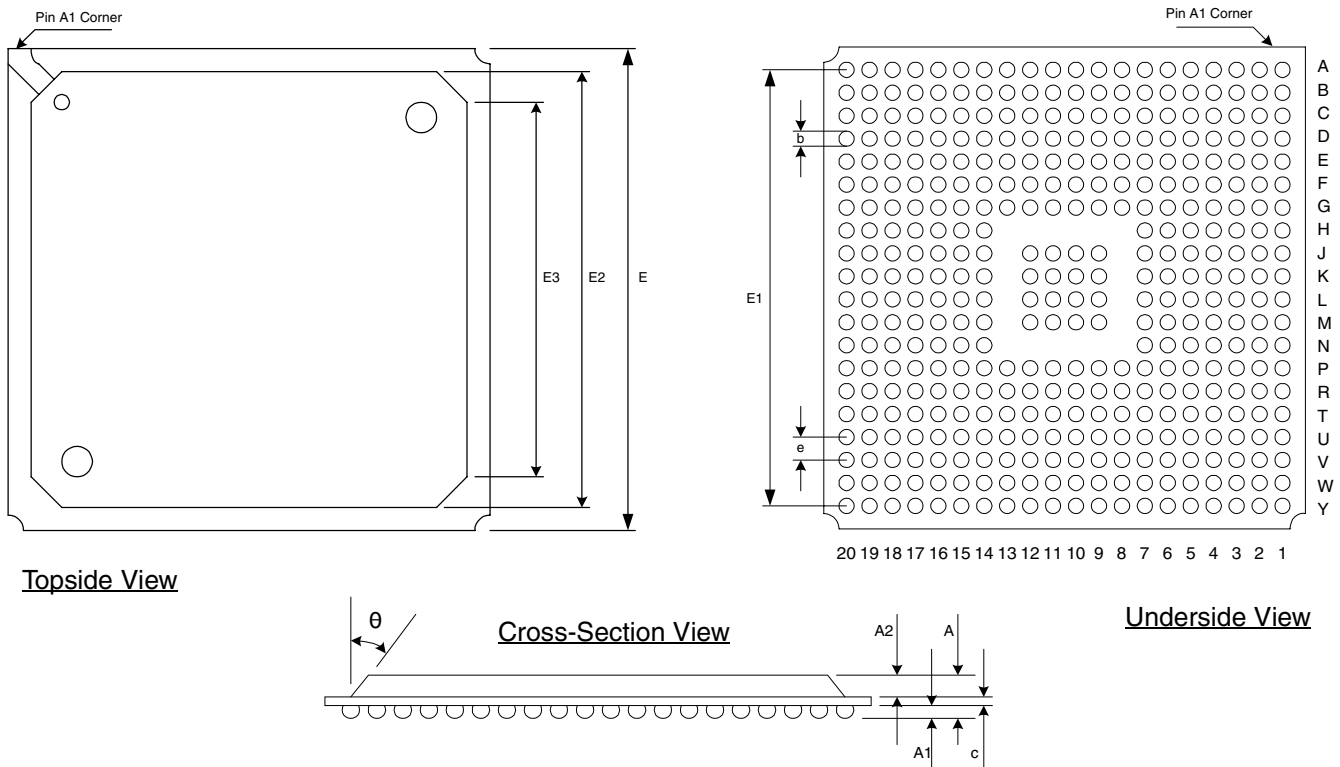


Figure 26-1. PCI 6540 Mechanical Dimensions

**Table 26-1. PCI 6540 Mechanical Dimensions**

Symbol	Dimension	Minimum	Nominal	Maximum
A	Overall package height	2.20	2.33	2.50
A1	Package standoff height	—	0.60	—
A2	Encapsulation thickness	1.12	1.17	1.22
b	Ball diameter	—	0.75	—
c	Substrate thickness	0.51	0.56	0.61
e	Ball pitch	—	1.27	—
E	Overall package width	26.80	27.00	27.20
E1	—	—	24.13	—
E2	Overall encapsulation width	23.80	24.00	24.20
E3	—	17.95	18.00	18.05
θ	—	—	30°	—

*This page intentionally left blank.*

## 26.2 PINOUT

	1	2	3	4	5	6	7	8	9	10
A	VSS	GPIO15	GPIO13	GPIO10	GPIO7	S_AD31	S_AD27	S_IDSEL	S_AD21	S_AD17
B	S_REQ0#	GPIO14	GPIO12	GPIO9	GPIO6	S_AD30	S_AD26	S_CBE3#	S_AD20	S_AD16
C	S_REQ2#	S_REQ1#	VDD_IO	GPIO8	GPIO5	S_AD29	S_AD25	S_AD23	VSS	S_CBE2#
D	S_REQ5#	S_REQ4#	S_REQ3#	GPIO11	GPIO4	S_AD28	S_AD24	S_AD22	S_AD19	S_FRAME#
E	S_GNT0#	ReserveIO1	S_REQ7#	S_REQ6#	VSS	S_PLLEN#	NC	VDD_CORE	S_AD18	S_IRDY#
F	S_GNT4#	S_GNT3#	S_GNT2#	S_GNT1#	S_CR	S_AVSS	S_AVDD	VDD_CORE	VDD_IO	VDD_IO
G	ReserveIO2	S_GNT7#	S_GNT6#	S_GNT5#	S_AVSS	S_AVDD	VDD_CORE	VDD_IO	VSS	VSS
H	S_XCAP_PU	S_RSTOUT#	S_RSTIN#	S_CFN#	VDD_CORE	VDD_CORE	VDD_IO			
J	MSK_IN	BPCC_EN	VSS	PRV_DEV (Transparent) XB_MEM (Non-Transparent)	S_CLKIN	VDD_IO	VSS		VSS	VSS
K	S_CLKO2	S_CLKO1	S_CLKO0	S_CLKOFF	S_CLKIN_STB	VDD_IO	VSS		VSS	VSS
L	P_RSTOUT#	P_RSTIN#	VDD_IO	S_CLKO4	S_CLKO3	VDD_IO	VSS		VSS	VSS
M	GPIO0	GPIO1	VSS	GPIO2	GPIO3	VDD_IO	VSS		VSS	VSS
N	OSCSSEL#	P_XCAP	PWRGD	P_CLKIN	S_XCAP_IN	VDD_CORE	VDD_IO			
P	P_REQ#	REFCLK	OSCIN	P_GNT#	P_AVSS	P_AVDD	VDD_CORE	VDD_IO	VSS	VSS
R	P_AD29	P_AD30	VDD_IO	P_AD31	P_CLKOE	P_AVSS	P_AVDD	VDD_CORE	VDD_IO	VDD_IO
T	P_AD25	P_AD26	P_AD27	P_AD28	VSS	P_CR	P_PLLEN#	VDD_CORE	P_DEVSEL#	P_SERR#
U	P_AD23	P_IDSEL	P_CBE3#	P_AD24	TMS	EJECT	P_BOOT	P_CBE2#	P_STOP#	P_PAR
V	P_AD21	P_AD22	VDD_IO	TDI	TRANS#	VDD_IO	P_AD18	P_FRAME#	VSS	P_CBE1#
W	P_AD19	P_AD20	TDO	DEV64#	U_MODE	L_STAT	P_AD17	P_IRDY#	P_LOCK#	P_AD15
Y	VSS	TCK	TRST#	EEPDATA	EEPClk	ENUM#	P_AD16	P_TRDY#	P_PERR#	P_AD14
	1	2	3	4	5	6	7	8	9	10

Figure 26-2. PCI 6540 Physical Layout with Pinout—Topside View (A1–A10 through Y1–Y10)

11	12	13	14	15	16	17	18	19	20	
S_TRDY#	S_SERR#	S_AD14	S_AD10	S_AD7	S_AD4	S_AD0	S_CBE7#	S_CBE5#	VSS	A
S_DEVSEL#	S_PAR	S_AD13	S_AD9	S_AD6	S_AD3	S_ACK64#	S_CBE6#	S_CBE4#	S_AD63	B
S_STOP#	VSS	S_AD12	S_AD8	VDD_IO	S_AD2	S_REQ64#	VDD_IO	S_AD61	S_AD62	C
S_LOCK#	S_CBE1#	S_AD11	S_CBE0#	S_AD5	S_AD1	S_AD57	S_AD58	S_AD59	S_AD60	D
S_PERR#	S_AD15	VDD_CORE	S_TST1	S_TST0	VSS	S_AD53	S_AD54	S_AD55	S_AD56	E
VDD_IO	VDD_IO	VDD_CORE	NC	TEST	NC	S_AD50	VDD_IO	S_AD51	S_AD52	F
VSS	VSS	VDD_IO	VDD_CORE	NC	NC	S_AD46	S_AD47	S_AD48	S_AD49	G
			VDD_IO	VDD_CORE	VDD_CORE	S_AD42	S_AD43	S_AD44	S_AD45	H
VSS	VSS		VSS	VDD_IO	S_AD38	S_AD39	VSS	S_AD40	S_AD41	J
VSS	VSS		VSS	VDD_IO	S_AD33	S_AD34	S_AD35	S_AD36	S_AD37	K
VSS	VSS		VSS	VDD_IO	S_M66EN	S_PME#	S_CLKRUN# (Transparent) S_INTA# (Non-Transparent)	S_PAR64	S_AD32	L
VSS	VSS		VSS	VDD_IO	P_PAR64	P_CLKRUN# (Transparent) P_INTA# (Non-Transparent)	VSS	P_PME#	P_M66EN	M
			VDD_IO	VDD_CORE	VDD_CORE	P_AD35	P_AD34	P_AD33	P_AD32	N
VSS	VSS	VDD_IO	VDD_CORE	NC	NC	P_AD39	P_AD38	P_AD37	P_AD36	P
VDD_IO	VDD_IO	VDD_CORE	NC	NC	NC	P_AD42	VDD_IO	P_AD41	P_AD40	R
P_AD13	P_AD8	VDD_CORE	P_TST1	P_TST0	VSS	P_AD46	P_AD45	P_AD44	P_AD43	T
P_AD12	P_CBE0#	P_AD5	P_AD1	P_CBE7#	P_CBE4#	P_AD60	P_AD54	P_AD48	P_AD47	U
P_AD11	VSS	P_AD4	P_AD0	VDD_IO	P_AD63	P_AD59	VDD_IO	P_AD51	P_AD49	V
P_AD10	P_AD7	P_AD3	P_ACK64#	P_CBE6#	P_AD62	P_AD58	P_AD55	P_AD52	P_AD50	W
P_AD9	P_AD6	P_AD2	P_REQ64#	P_CBE5#	P_AD61	P_AD57	P_AD56	P_AD53	VSS	Y
11	12	13	14	15	16	17	18	19	20	

Figure 26-3. PCI 6540 Physical Layout with Pinout—Topside View (A11–A20 through Y11–Y20)





# A USING PCI 6540

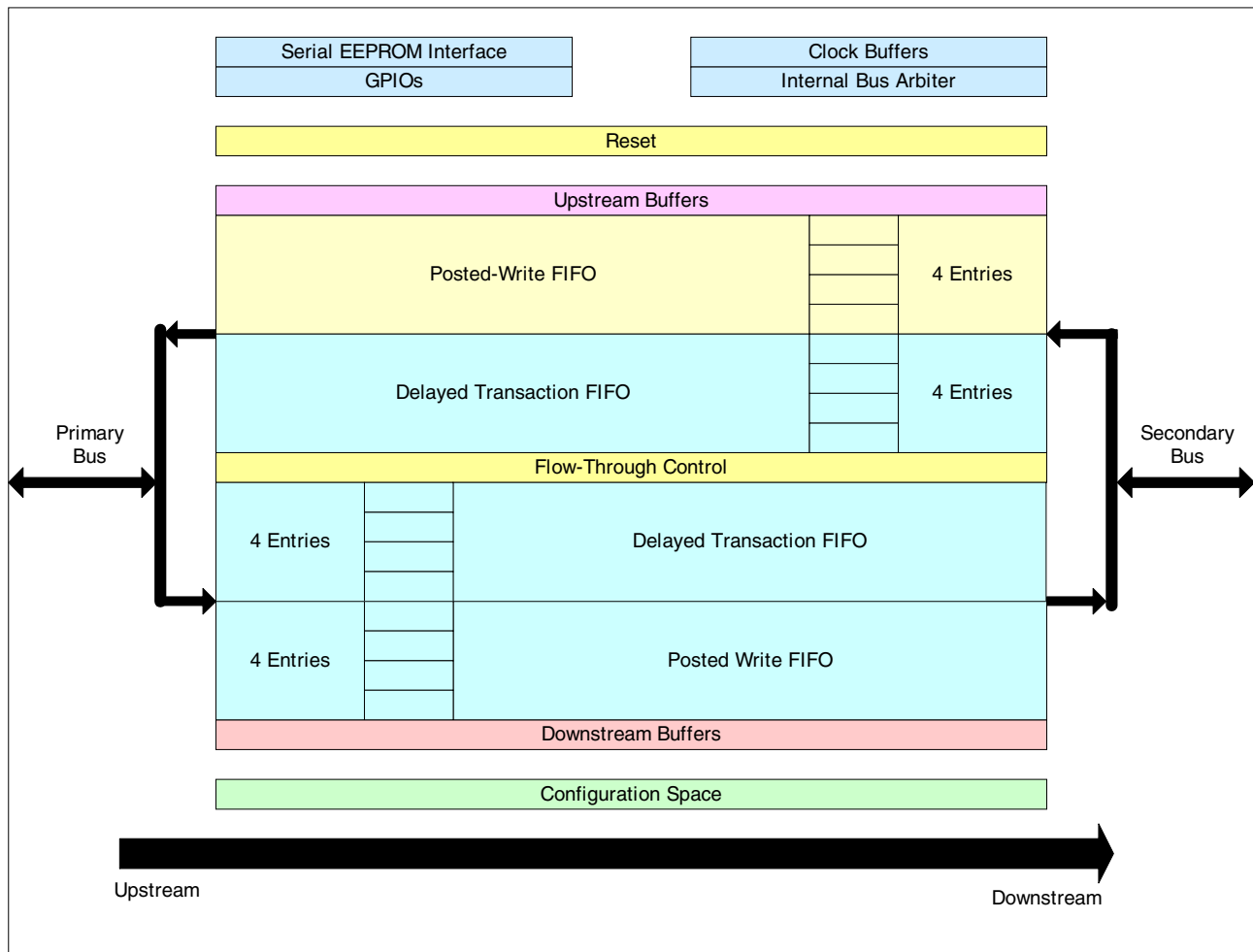


Figure A-1. PCI 6540 Internal Architecture

## A.1 TRANSPARENT MODE APPLICATION

Because the PCI 6540 primary and secondary ports are asynchronous to one another, these two independent systems can run at differing frequencies. The secondary bus may be run faster than the primary bus, and vice versa. The PCI 6540 can be set to enforce PCI-X protocol, without requiring standard PCI-X reset initialization.

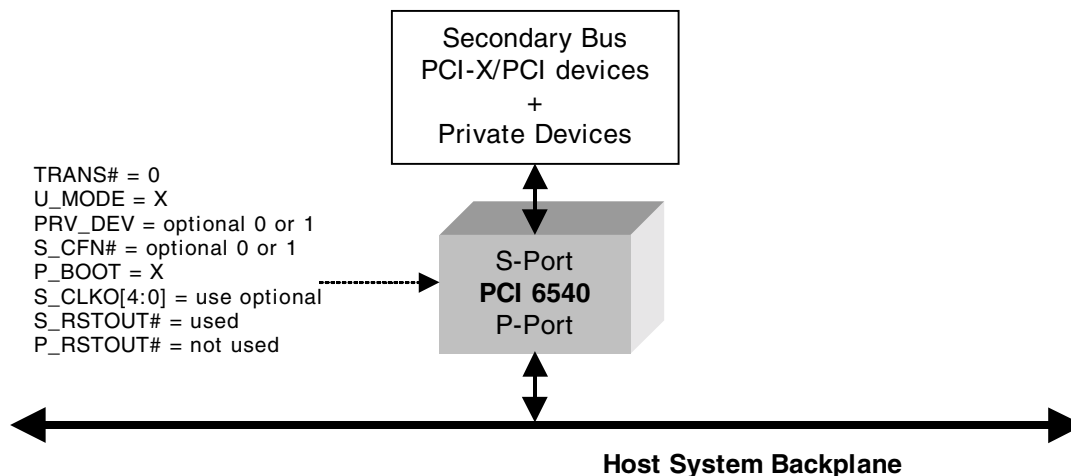
The PCI 6540 controls powerful programmable buffers, which can be used to regulate data throughput for multiple PCI masters on the secondary port. The FIFO can be divided into four independent segments, each dedicated to its corresponding entry. The PCI 6540 can be programmed to prefetch up to 2 KB at a time and the data can be stored in the FIFO without being flushed until timed out. This allows the PCI 6540 to truly prefetch data on behalf of PCI devices, with minimum bus bandwidth requirement on the PCI-X Bus.

In Transparent mode, the host system PCI-X Bus is connected to the PCI 6540 primary port. The secondary PCI port can use a custom-designed External Arbiter or the PCI 6540 Internal Arbiter. Use the custom-designed clock generations, PCI 6540 S\_CLKO[4:0] outputs (derived out of the primary port PCI clock input), or an external oscillator (to provide clocks to secondary PCI devices and PCI 6540 S\_CLKIN).

The PCI 6540 also supports Private PCI devices on the secondary bus. By setting PRV\_DEV to 1, the PCI 6540 allows secondary port IDSEL rerouting, using S\_AD[23:16] to S\_AD24. When S\_AD24 has no device connected to it, S\_AD[23:16] Type 1 Access cycles are Master Aborted.

By setting PRV\_DEV to 1, and programming the corresponding special Memory Range registers, the PCI 6540 also reserves a Private Memory region for secondary port private device use only. The PCI 6540 does not respond to accesses to this private region by primary or secondary PCI masters.

Figure A-2 provides basic optimization design.



**ENFORCE PCI-X: PCI 6540 has a P\_XCAP input pin to enforce PCI-X protocol without requiring standard PCI-X reset initialization**

Figure A-2. PCI 6540 Transparent Mode Basic Optimization Design

## A.2 NON-TRANSPARENT MODE APPLICATION

The PCI 6540 Non-Transparent mode acts as a memory-mapped PCI-X device on a PCI-X backplane. The PCI 6540 primary bus is used to connect to the PCI-X backplane, *such as* for Transparent mode applications. The intelligent subsystem is connected to the secondary port. The subsystem can use another External Arbiter or the PCI 6540 Internal Arbiter. Subsystem clock generation is generally achieved using a clock synthesizer to provide CPU clocks, subsystem PCI-X clocks to subsystem PCI-X devices and the PCI 6540 S\_CLKIN. The subsystem can also use the PCI 6540 S\_CLKO[4:0] outputs derived from the primary port PCI-X clock input or an external oscillator.

Program the P\_BOOT pin to 0, so the subsystem at the secondary port retains higher boot priority. The subsystem must set up the BARs first, or the XB\_MEM option must be active, for the primary port host to be able to complete its system initialization sequence. Use of the XB\_MEM option forces the PCI 6540 to declare a fixed 16 MB memory window for cross-bridge communication at power-up. If necessary, this window size can be changed by the serial EEPROM or software after power-up.

Primary and secondary ports retain independent PCI-X reset inputs. Custom-designed reset or PCI 6540 S\_RSTOUT# can be used for the secondary port and subsystem reset.

Figure A-3 provides basic optimization design.

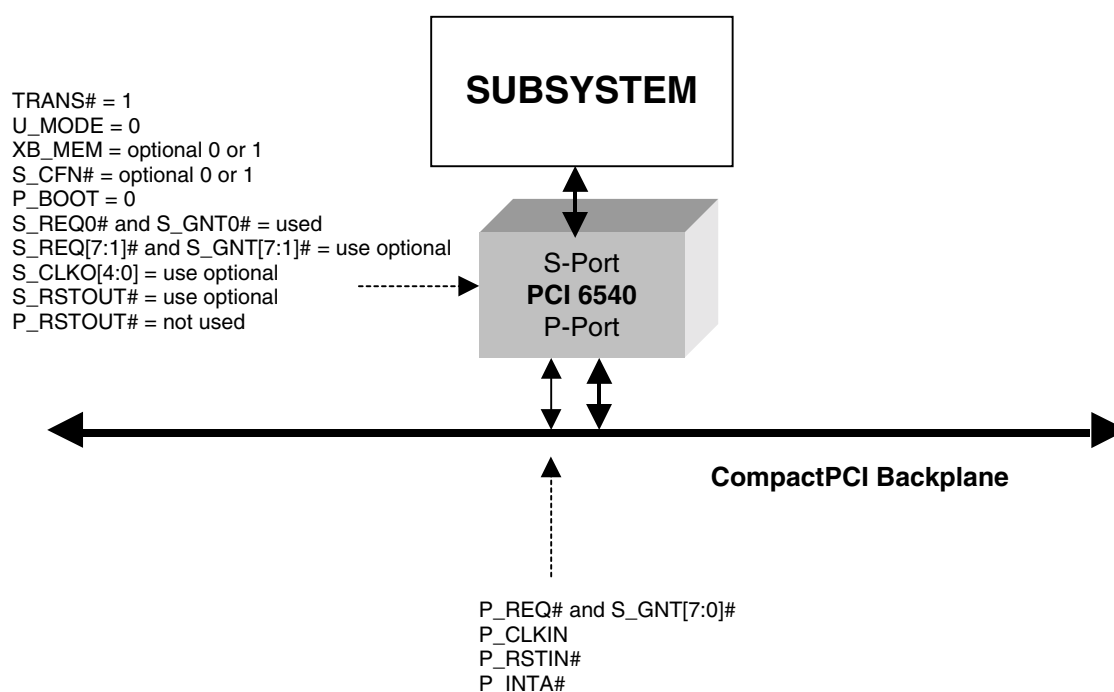


Figure A-3. PCI 6540 Non-Transparent Mode Basic Optimization Design

### A.3 UNIVERSAL BRIDGING APPLICATION

The PCI 6540 is designed to allow an intelligent subsystem design to operate as a host or memory-mapped device by setting the U\_MODE (Universal mode) pin to 1.

When operating as a host, the intelligent subsystem uses Universal Transparent mode. The subsystem PCI-X Bus is connected to the PCI 6540 primary port. The subsystem uses an External Arbiter or an Arbiter built-in to the north bridge for subsystem PCI-X Bus support. The backplane PCI-X Arbiter can use a custom-designed Arbiter or the PCI 6540 Internal Arbiter. Subsystem clock generation is generally achieved using a clock synthesizer to provide CPU clocks, Subsystem PCI-X clocks to subsystem PCI-X devices, and the PCI 6540 P\_CLKIN. Use custom-designed clock outputs, the PCI 6540 S\_CLKO[4:0] outputs derived out of the primary port PCI-X clock input, or an external oscillator (to drive the PCI-X backplane).

When operating as an intelligent subsystem, behaving as a memory-mapped PCI-X device on the backplane PCI-X Bus, the subsystem uses Universal Non-Transparent mode. The PCI 6540 External Arbiter mode is selected so that S\_REQ0# and S\_GNT0# function as PCI\_REQ# and PCI\_GNT#, respectively, for direct connection to the backplane or custom-designed Arbiter interface. Connect the P\_BOOT pin to 1, indicating that the primary port has boot priority. The subsystem at the primary port must set up the BARs first, or the XB\_MEM option must be active for the host from the secondary port to be able to complete the system initialization sequence. Use of the XB\_MEM option forces the PCI 6540 to declare a fixed 16 MB Memory window for cross-bridge communication at power-up. If necessary, the window size can be changed by the serial EEPROM or software after power-up.

Figure A-4 illustrates a basic optimization design, assuming the same CPU board is behaving differently.

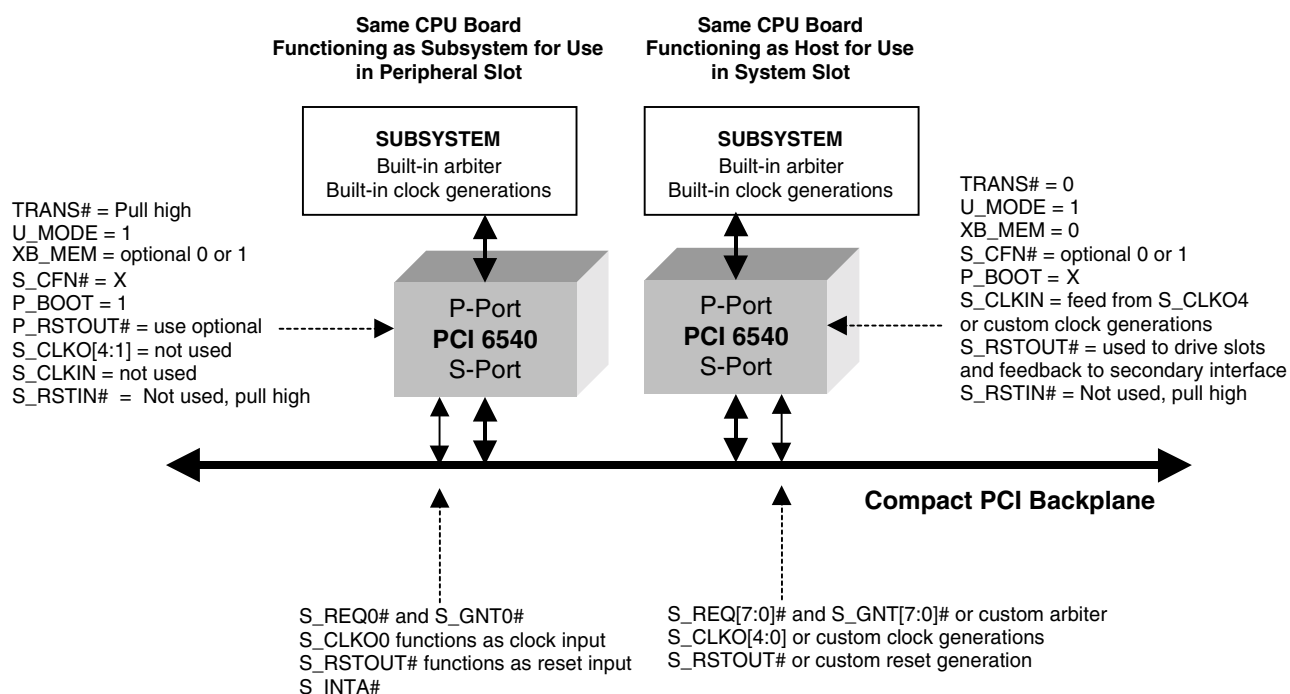


Figure A-4. PCI 6540 Universal Bridging Application Basic Optimization Design

### A.3.1 Universal Mode CLK, RST#, REQ0#, GNT0#, and SYSEN# Signal Connections

The PCI 6540 allows a jumperless automatic switch between CompactPCI system slot or peripheral slot applications. In Universal mode, the PCI 6540 switches between System and Peripheral mode using the TRANS# input pin, which is designed for direct connection to the CompactPCI SYSEN# pin.

When TRANS# is 0, the PCI 6540 switches to Universal Transparent mode, in which the PCI 6540 drives out S\_RSTOUT# to the backplane, and enables use of the S\_RSTOUT# internal feedback for Secondary reset. Secondary port logic uses S\_CLKIN. For example, use S\_CLKO4 to feed S\_CLKIN with a clock trace length that matches the backplane clock traces length. If a custom-designed Arbiter is not used,

S\_REQ0# and S\_GNT0# can be directly connected to the backplane.

When TRANS#=1, the PCI 6540 switches to Universal Non-Transparent mode, in which the PCI 6540 places S\_RSTOUT# into a high-impedance state to allow the backplane RST# signal to drive the S\_RSTOUT# pin (which acts as secondary Reset input). The S\_CLKIN pin is ignored inside the PCI 6540 and secondary port logic uses the S\_CLKO0 internal feedback as a secondary Clock input. S\_CFN# is *Don't Care* and if a custom-designed Arbiter is not used, S\_REQ0# and S\_GNT0# can be directly connected to the backplane.

*Hot Swap pin ENUM#* must be controlled with custom logic for universal applications.

Figure A-5 and Table A-1 provide an example of Universal mode connections. In this example, U\_MODE=1.

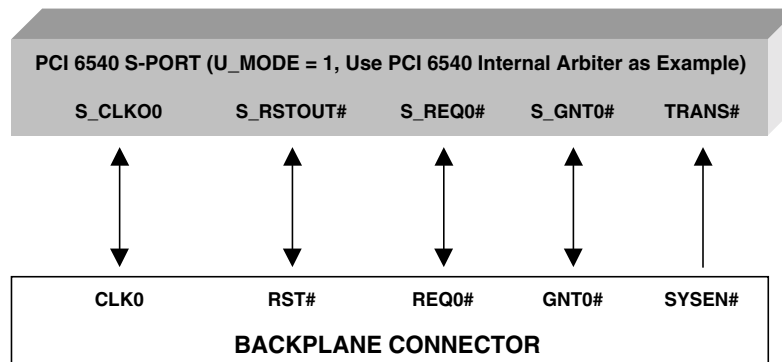


Figure A-5. Universal Mode Connections Example

Table A-1. Universal Mode Connection Example

Description	PIN									
	TRANS# Connected to CompactPCI SYSEN# Pin	U_MODE	S_CFN#	S_CLKO0	S_CLKIN	S_CLKO4	S_RSTIN#	S_RSTOUT#	S_REQ0#	S_GNT0#
System Slot application using PCI 6540 Internal Arbiter	0 Use S_CLKIN as secondary port input clock	1	0	OUTPUT	INPUT, such as from S_CLKO4	OUTPUT	<i>Not Used</i>	OUTPUT	REQ0# INPUT	GNT0# OUTPUT
System Slot application using External Arbiter	0 Use S_CLKIN as secondary port input clock	1	1	OUTPUT	INPUT, such as from S_CLKO4	OUTPUT	<i>Not Used</i>	OUTPUT	Custom Arbiter Backplane REQ# OUTPUT	Custom Arbiter Backplane GNT# INPUT
Peripheral Slot application using PCI 6540 Universal mode	1 Use S_CLKO0 as secondary port input clock	1	X	Used as secondary Clock INPUT	Ignored in PCI 6540 <i>Not Used</i>	OUTPUT <i>Not Used</i>	<i>Not Used</i>	Used as secondary Reset INPUT	Backplane REQ# OUTPUT	Backplane GNT# INPUT

## A.4 SYMMETRICAL NON-TRANSPARENT APPLICATION

The PCI 6540 is designed to allow the bridging of two totally independent systems. The only required option is to decide which host has higher boot priority.

The PCI 6540 External Arbiter mode is selected so that S\_REQ0# and S\_GNT0# function as PCI GNT# and PCI-X REQ#, respectively, for handshaking with one of the host. All PCI signals should be connected to their respective host PCI-X signals. Boot priority is programmable. The higher priority boot system must set up the BARs first, or the XB\_MEM option must be active, for the lower boot priority system to be able to complete its system initialization sequence. Use of the

XB\_MEM option forces the PCI 6540 to declare a fixed 16 MB memory window for cross-bridge communication at power-up. If necessary, this window size can be changed by the serial EEPROM or software after power-up.

The independent system uses an External Arbiter or the north bridge Internal Arbiter for PCI-X Bus use. System clock generation is generally achieved using a clock synthesizer to provide CPU clocks, PCI-X clocks to system PCI-X devices and the PCI 6540 PCI-X input.

Because the PCI 6540 primary and secondary ports are asynchronous to one another, the two independent systems can run at differing frequencies.

Figure A-6 provides basic optimization design.

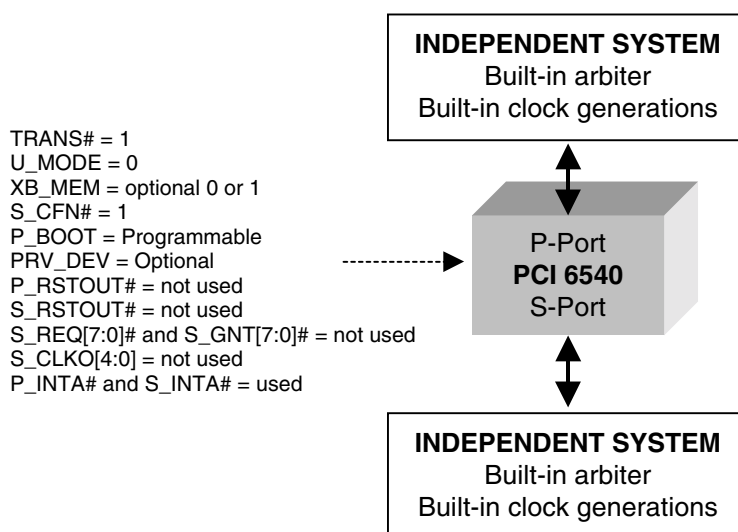


Figure A-6. PCI 6540 Symmetrical Non-Transparent Application Basic Optimization Design





## B PCI-X CLOCK AND FREQUENCY INITIALIZATION SEQUENCE

### B.1 BUS SPEED AND TYPE DETECTION

Figure B-1 provides an example of a CPLD being used to detect bus type and speed and feed to the PCI 6540 during the Reset phase.

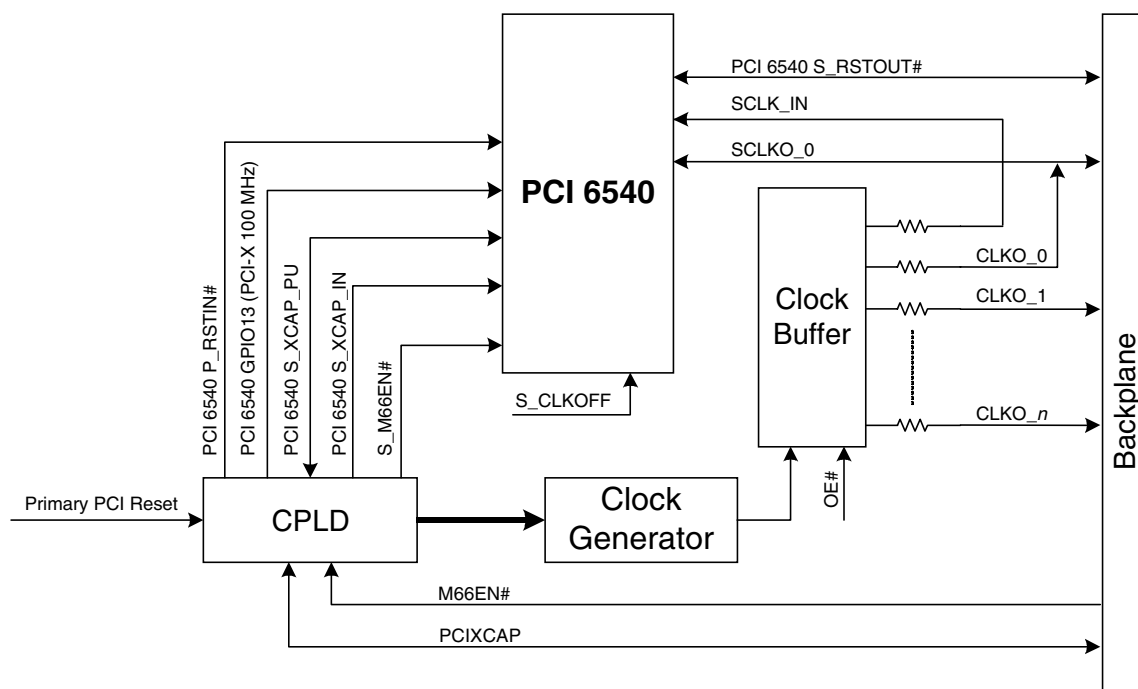
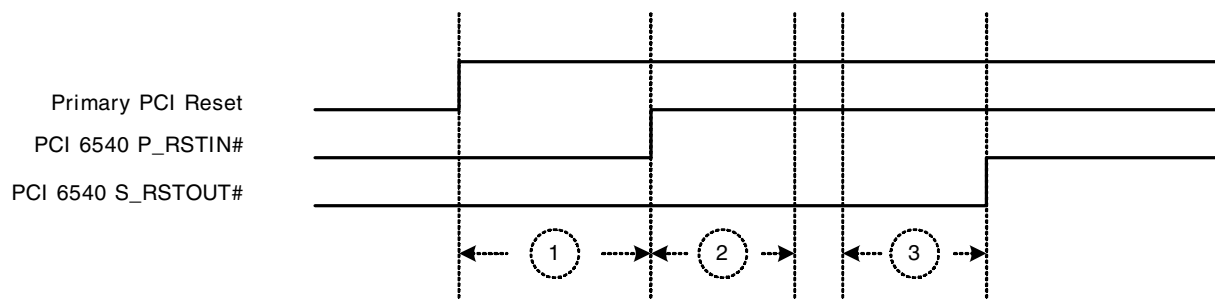


Figure B-1. CPLD Used to Detect Bus Type and Speed and Feed to PCI 6540 during Reset Phase

### B.1.1 Universal Transparent Mode

Because of the differences between standard PCI-X and CompactPCI PCI-X Bus speed and bus type detection scheme, a custom programmable logic is required to detect CompactPCI PCI-X speed. After detection, such custom logic is needed to emulate the standard PCI-X Bus speed and bus type feedback to the PCI 6540. Figure B-2 provides an example of this process.



Stage 1: CPLD acquires bus speed information from Backplane  
 Stage 2: PCI 6540 acquires bus speed information from CPLD  
 Stage 3: PCI 6540 reports bus speed information to PCI Bus in Backplane

Figure B-2. Bus Mode Detecting Sequence

### B.1.2 Universal Non-Transparent Mode

The PCI 6540 detects PCI-X mode from the PCI Bus when S\_RSTOUT# is de-asserted during a cold boot.

## B.2 SECONDARY CLOCK OUTPUTS

**Note:** Secondary clocks from the PCI 6540 are not recommended for PCI-X use. Use high-quality clock buffers for PCI-X applications.

A high-quality clock generator and clock buffer should be used for 100 and 133 MHz PCI-X applications.

All clocks must have similar flight time, no matter whether they are connected to secondary port PCI-X devices or used for feedback to the PCI 6540 (*that is*, all clock traces are to be of similar length or are delay compensated).

### B.3 INTERNAL CLOCK DIVIDER

The PCI 6540 internal clock divider works with PCI only applications. **For PCI-X applications, do not use the divider.**

### B.4 PCI-X IN UNIVERSAL MODE S\_CLKO0 PIN USE

The PCI 6540 uses different clock inputs in a system slot than those it uses in a peripheral slot. In system slot applications, the PCI 6540 uses S\_CLKIN as the clock source for all internal logic. In peripheral slot applications, the PCI 6540 uses the S\_CLKO0 pin as the clock input source for all internal logic.

# C PCI 6520BB AND PCI 6540BB PIN COMPARISON

## C.1 OVERVIEW

The following is extrapolated from *PCI 6520BB and PCI 6540BB Pin Comparison Version 1.3*, an application note, published October 30, 2003, by PLX Technology, Inc.

**Table C-1. Part Number Conversion**

HiNT Part Number	PLX Part Number
HB7-AB	PCI6520-BB13BC
HB8-BB	PCI6540-BB13BC

**Table C-2. PCI 6520BB Versus PCI 6540BB Pin Assignment Comparison**

Pin Location	PCI 6520BB	PCI 6540BB
U6	Reserved V <sub>SS</sub>	EJECT
Y6	NO CONNECTION	ENUM#
A4	Reserved IO3	GPIO10
D4	Reserved IO4	GPIO11
B3	Reserved IO5	GPIO12
A3	PCI-X 100 MHz <sup>1</sup>	GPIO13 / PCI-X 100 MHz
B2	Reserved IO6	GPIO14
A2	Reserved IO7	GPIO15
C4	Reserved IO8	GPIO8
B4	Reserved IO9	GPIO9
W6	Reserved IO10	L_STAT
U7	NO CONNECTION	P_BOOT
M19	NO CONNECTION	P_PME#
L17	NO CONNECTION	S_PME#
L1	NO CONNECTION	P_RSTOUT#
N2	Reserved V <sub>SS</sub>	P_XCAP
A8	Reserved V <sub>SS</sub>	S_IDSEL

Table C-2. PCI 6520BB Versus PCI 6540BB Pin Assignment Comparison (Continued)

Pin Location	PCI 6520BB	PCI 6540BB
H3	Reserved V <sub>DD_IO</sub>	S_RSTIN#
V5	Reserved V <sub>SS</sub>	TRANS#
W5	Reserved V <sub>SS</sub>	U_MODE
F15	NO CONNECTION	TEST

1. Revision 1.0 of this document mistakenly listed PCI 6520 pin A3 as NO CONNECTION.

## D GENERAL INFORMATION

The PCI 6540 is a 64-bit, 133 MHz PCI-X-to-PCI-X bridge that supports both transparent and non-transparent operation. The PCI 6540 provides a range of added-value features to system designers, including:

- Two PCI-X ports, each capable of running at the full 64-bit, 133 MHz speed
- Asynchronous primary and secondary ports
- Universal mode operation, allowing the same card to be both a system card or a peripheral card in a CompactPCI chassis
- 5V tolerant I/O
- Programmable prefetch
- Programmable Flow Through
- Zero wait state burst

- 10-KB Data FIFO
- Five secondary clock outputs
- Reference clock input for frequency detection
- PCI Power Management support
- Arbitration support for eight secondary bus masters
- Serial EEPROM for configuration
- 16 General Purpose I/O pins
- CompactPCI Hot Swap *Ready*
- Vital Product Data (VPD) support
- JTAG boundary scan

The PCI 6540 is offered in a 380-pin (ball) PBGA, and is designed to operate over the Commercial Temperature range.

## D.1 ORDERING

### Table D-1. Available Package

Package	Ordering Part Number
380-pin PBGA	PCI6540-BB13BC

**PCI 6540-BB13BC**

```

graph LR
    A[PCI 6540-BB13BC] --- B[PCI 6540]
    A --- C[BB]
    A --- D[13]
    A --- E[B]
    A --- F[C]
    A --- G[ES]
        
```

BB—Part Revision Code  
 13—Speed Grade (133 MHz PCI-X Bus)  
 B—Package Type  
     B = Plastic Ball Grid Array  
 C—Case Temperature  
     I = Industrial Temperature  
     C = Commercial Temperature  
     ES = Engineering Sample  
 PCI 6540—Family/Core PCI 6540 device

## **D.2 UNITED STATES AND INTERNATIONAL REPRESENTATIVES, AND DISTRIBUTORS**

A list of PLX Technology, Inc., representatives and distributors can be found at <http://www.plxtech.com>.

## **D.3 TECHNICAL SUPPORT**

PLX Technology, Inc., technical support information is listed at <http://www.plxtech.com/support/>, or call 408 774-9060 or 800 759-3735.

# Index

## A

abnormal termination 17-2

abort

- master 3-5, 3-7, 3-9, 3-12, 3-25, 6-5, 6-10, 6-15, 6-16, 6-32, 6-34, 6-64, 6-74, 6-82, 6-103, 6-104, 8-9, 8-11, 8-12, 8-12-8-13, 8-14, 8-16, 8-18, 9-1, 9-2, 9-4, 9-9, 9-11, 9-16, 9-17, 9-18, 10-5, 12-6, 13-2, 17-1, 17-2, A-2
- target 3-7, 3-12, 6-5, 6-10, 6-15, 6-32, 6-34, 6-64, 6-73, 6-74, 6-82, 6-103, 6-104, 8-3, 8-4, 8-7, 8-12, 8-13, 8-14, 8-15, 8-16, 8-18, 9-3, 9-4, 9-8, 9-9, 9-16, 9-18, 9-19, 12-1, 12-2, 12-4, 12-6, 13-2

abort, target 9-17

access, exclusive 13-1-13-2

ACNTRL register 6-17, 6-28, 6-128, 14-1, 14-2

ADB 9-4-9-21, 19-2

address decoding 10-1-10-12

Address Translation Control registers 6-42-6-50, 6-120-6-126, 10-5, 10-11, 20-1

Allowable Disconnect Boundary

- See* ADB

Arbiter Control registers 6-2, 6-16-6-17, 6-99, 6-127-6-128, 14-1

arbitration, PCI 14-1-14-3

architectural boundary scan

- See* IEEE Standard

Attributes 9-1, 9-2, 9-3, 9-4, 9-6-9-7, 9-8, 9-9, 9-11, 9-11-9-13, 9-14, 9-19-9-20

- Relaxed Ordering bit 11-4, 11-6

## B

BCNTRL register 5-8, 6-14-6-15, 6-17, 6-19, 6-82, 6-128, 19-3

Boundary Scan

- Description Language 24-2
- pins 3-2, 3-18

BPCC\_EN 3-21, 5-9

bridge

- behavior 17-1-17-3
- Control register 6-2, 6-14-6-15, 6-81, 6-82-6-83
- PCI 6000 series 1-2-1-3
- PCI-X Status register 6-3, 6-58, 6-99, 6-135
- Supports Extension register 6-3, 6-53, 6-99, 6-131

BSDL 24-2

BUFCR register 6-27, 6-93, 19-3, 19-4

Buffer Control register 6-2, 6-27, 6-81, 6-93

buffers

- ADB size 9-4
- I/O 24-2
- multiple write transactions 8-5

bus operation

- PCI 8-1-8-18
- PCI-X 9-1-9-21

## C

CAP\_PTR register 6-5, 6-14, 6-64, 6-70, 6-73, 6-79

CCNTRL register 3-25, 6-16, 6-36, 6-81, 6-117, 6-127, 8-5, 10-5

Chip Control registers 3-25, 6-2, 6-16-6-17, 6-36, 6-81, 6-99, 6-127-6-128, 8-5

CLKCNTRL register 3-2, 3-13, 3-15, 4-1, 5-6, 6-34, 6-102

CLKRUN register 6-35

Clock Control register 6-102

clock, PCI-X B-1-B-2

clocking 4-1-4-4

Clock-Related pins 3-2, 3-13-3-15

commands 16-1-16-4

- conventional PCI-to-PCI-X 9-20
- PCI-X register 6-134
- PCI-X-to-conventional PCI 9-21
- primary 3-5, 6-2, 6-61
- Primary PCI register 6-4, 6-63
- read queue 2-2
- secondary 3-8, 3-9, 6-61
- Secondary PCI register 6-72
- serial EEPROM 6-95, 7-1

CompactPCI Hot Swap

- See* Hot Swap

completion

- delayed read 8-7-8-8
- delayed write 11-2
- split 9-1-9-19

Control registers 2-1, 6-16-6-17, 6-127-6-128, 8-6

controller, test access port (TAP)

- See* test access port controller

cross-bridge

- Configuration Access Control registers 6-100-6-101
- memory window enable 3-25

## D

DAC 3-4, 3-8, 8-1, 8-2, 9-2, 9-5, 9-9, 9-10, 9-19, 16-2, 16-4

DCNTRL register 3-16, 5-7, 5-8, 6-17, 6-128

deadlock 11-1, 11-2, 11-4-11-6

debug 24-1-24-2

decoding 10-1-10-12

decoupling, power supply 3-3

delayed write 3-7, 3-12, 6-20, 6-32, 6-34, 6-86, 6-103, 6-104, 6-105, 8-2, 8-4, 8-5, 8-12, 8-14, 8-17, 8-18, 11-1, 11-2, 11-3, 12-3, 12-5, 12-6, 12-8, 12-10, 12-12

## DEV64#

3-21, 5-9, 6-29, 6-58, 6-95, 6-135  
device hiding 5-9–5-13, 6-54, 6-132, 22-2  
Diagnostic Control registers 3-16, 5-6, 5-7, 6-16–6-17,  
6-127–6-128  
Direct Message Interrupt registers 5-6, 6-110–6-111,  
20-2  
Doorbell Interrupt registers 6-99, 6-114–6-116, 20-1,  
20-2  
Dual Address Cycle  
    See DAC  
DWNBAR0MSK register 6-49, 6-67, 6-125, 7-4, 10-9,  
10-11  
DWNBAR1MSK register 6-43, 6-48, 6-49, 6-68, 6-69,  
6-124, 6-125, 7-4, 10-9, 10-11  
DWNBAR2MSK register 6-50, 6-69, 6-125, 7-4, 10-9,  
10-11  
DWNDBIE register 6-114  
DWNDBIR register 6-114  
DWNDBIS register 6-115  
DWNINTE register 5-6, 6-116  
DWNINTSR register 5-6, 6-111, 6-115  
DWNMSG0 register 6-111  
DWNMSG1 register 6-111  
DWNMSG2 register 6-111  
DWNMSG3 register 6-111  
DWNTNBAR0 register 6-48, 6-124, 7-4, 10-12  
DWNTNBAR1 register 6-48, 6-124, 10-9, 10-12  
DWNTNBAR2 register 6-48, 6-124, 10-9, 10-12  
DWNTNE register 6-50, 6-126, 7-4, 10-10, 10-12, 20-1

## E

Early Power Support 22-1  
ECP 21-1  
EEPADDR register 6-30, 6-96, 7-1  
EEPCLK 3-18, 5-9, 7-1  
EEPCNTRL register 6-29, 6-95, 7-1  
EEPDATA pin 3-18, 5-9, 7-1  
EEPDATA register 6-30, 6-96, 7-1  
EJECT 3-17, 5-9, 22-1  
electrical specs 25-1  
Enhanced Capabilities Port  
    See ECP  
ENUM# 3-17, 5-9, 6-54, 6-132, 22-1, 22-2, A-5  
error handling 12-1–12-13  
exclusive access 13-1–13-2  
Extended registers 5-2, 5-14, 6-3, 6-40, 6-45, 6-99,  
6-118, 6-119–6-120, 10-6, 10-7  
EXTRDATA register 5-14, 6-40, 6-41, 6-67–6-69, 6-76–  
6-78, 6-119, 6-120, 10-6  
EXTRIDX register 5-14, 6-40, 6-41, 6-67–6-69, 6-76–  
6-78, 6-119, 6-120, 10-6

## F

FIFO 19-1–19-4  
fixed priority scheme 14-2–14-3  
flow-through 18-1–18-3  
    primary 6-2, 6-18, 6-81, 6-82–6-84, 7-3, 18-1, 18-2  
    secondary 6-2, 6-26, 6-81, 6-92, 7-3

## G

GPIO 15-1–15-2  
    pins 3-3, 3-19–3-20, 22-2  
GPIO[15:14, 12:8] pins 3-19  
GPIO[15:8] registers 6-108–6-109  
GPIO[3:0] pins 3-19  
GPIO[3:0] registers 6-106  
GPIO[7:4] pins 3-19  
GPIO[7:4] registers 6-108–6-109  
GPIO13 3-20, 5-1, 15-2  
GPIOID[15:8] register 6-39, 6-109  
GPIOID[3:0] register 6-33, 6-106, 15-1  
GPIOID[7:4] register 6-38, 6-108, 15-1  
GPIOOD[15:8] register 6-39, 6-109  
GPIOOD[3:0] register 6-33, 6-106, 15-1  
GPIOOD[7:4] register 6-38, 6-108, 15-1  
GPIOOE[15:8] register 3-19, 3-20, 6-39, 6-109  
GPIOOE[3:0] register 3-19, 6-33, 6-106, 15-1  
GPIOOE[7:4] register 3-19, 6-38, 6-108, 15-1

## H

hardware 24-1  
Header registers 6-4–6-15, 6-62–6-71, 6-72–6-80  
Hot Swap 3-1, 22-1–22-2  
    pins 3-17  
    registers 3-17, 6-37, 6-54, 6-107, 6-132, 22-1, 22-2  
HS\_CNTL register 3-17, 6-54, 6-132  
HS\_CSR register 3-17, 6-54, 6-132, 22-1, 22-2  
HS\_NEXT register 3-17, 6-54, 6-132  
HSSRRC register 3-17, 6-3, 6-37, 6-51, 6-107, 10-6,  
10-9, 10-10, 14-3

## I

IACNTRL register 6-28, 6-94, 7-3, 14-1, 14-2, 14-3  
IEEE Standard 1149.1-1990 24-1–24-2  
*IEEE Standard Test Access Port and Boundary-Scan  
Architecture*  
    See IEEE Standard 1149.1-1990  
incremental prefetch count 6-2, 6-23, 6-24, 6-81, 6-89,  
6-90, 7-3, 18-3  
initialization  
    5-1–5-14



## interface

- debug 24-1–24-2
- GPIO 15-1–15-2
- high availability 22-1
- JTAG 24-1–24-2
- primary 12-7, 16-1–16-2, 20-1, 25-3
- secondary 12-7, 16-3–16-4, 20-1, 25-3

## Internal Arbiter Control register

- 6-2, 6-27, 6-28, 6-81, 6-93, 6-94, 7-3, 14-1, 14-2

## Interrupt registers 6-110–6-116, 20-2

## ISA 6-14, 6-82, 10-1, 10-4

## J

### JTAG 24-1–24-2

- pins 3-2, 3-18

## L

### L\_STAT 3-17, 5-9, 22-1

### locks 13-1–13-2

## M

- master abort 3-5, 3-7, 3-9, 3-12, 3-25, 6-5, 6-10, 6-15, 6-16, 6-32, 6-34, 6-64, 6-74, 6-82, 6-103, 6-104, 8-9, 8-11, 8-12, 8-12–8-13, 8-14, 8-16, 8-18, 9-1, 9-2, 9-4, 9-9, 9-11, 9-16, 9-17, 9-18, 10-5, 12-6, 13-2, 17-1, 17-2, A-2

### mechanical specs 26-1–26-4

### memory

- prefetchable 6-12–6-13, 10-3
- private 6-2, 6-16, 6-36, 10-5, A-2
- write and invalidate 6-4, 6-6, 6-21, 6-63, 6-66, 6-72, 6-75, 6-87, 8-1, 8-2, 8-3, 8-12, 8-15, 9-20, 11-1, 16-2, 16-4

### Message Signaled Interrupt registers 6-112–6-113, 20-2

### Miscellaneous Options register 6-2, 6-20–6-21, 6-81, 6-86–6-87, 7-3, 8-3, 8-9, 17-3

### Miscellaneous pins 3-3, 3-21–3-23

### MSCOPT register 6-20–6-21, 6-86–6-87, 7-3, 8-3, 8-9, 11-2, 13-2, 17-3

### MSIADDR register 6-113

### MSIC register 6-112

### MSICAPID register 6-107, 6-112

### MSIDATA register 6-113

### MSINEXT register 6-112

### MSIUADDR register 6-113

### MSK\_IN 3-2, 3-13, 5-9

### Multiplexed pins 3-1, 3-3, 3-25–3-26

## N

### NC 3-24

### No Connect pins 3-1, 3-24, C-1

### Non-Transparent Configuration Ownership Semaphore register 6-62, 6-98, 6-99, 6-117

### Non-Transparent mode 2-2, 20-1, A-3

- pin states 5-9–5-13
- pins 3-25–3-26
- registers 6-60–6-136
- reset inputs effect 5-8
- symmetrical application A-7
- timing 5-5

*See Also* Universal Non-Transparent mode

### normal termination vs. master abort 8-12, 8-13

### NTCOS register 6-62, 6-117

## O

### optimization

- basic design A-2–A-4, A-7
- flow-through 18-1–18-3

### ordering, transactions 11-1–11-6

### OSCIN 3-2, 3-13, 4-1, 4-2, 5-2, 5-9

### OSCSSEL# 3-2, 3-13, 4-1, 4-2, 5-9

## P

### P\_ACK64# 3-1, 3-4, 5-9

### P\_AD[31:0] 3-4, 16-2

### P\_AD[63:0] 3-1

### P\_AV<sub>DD</sub> 3-24, 5-9

### P\_AV<sub>SS</sub> 3-24, 5-9

### P\_BOOT 3-21, 5-6, 5-7, 5-10, 21-1

### P\_CBE[3:0]# 3-5, 5-10, 8-2, 8-9, 16-1–16-2

### P\_CBE[7:0]# 3-1

### P\_CBE[7:4]# 3-5, 5-10, 8-2, 8-9

### P\_CLKIN 3-2, 3-13, 4-1, 4-4, 5-2, 5-6, 5-7, 5-9, 5-10, 22-1

### P\_CLKOE 3-2, 3-13, 4-1, 5-10, 6-29, 6-95

### P\_CLKRUN# 3-26, 5-10

### P\_CR 3-2, 3-13, 5-10

### P\_DEVSEL# 3-1, 3-5, 5-10, 8-8, 12-1

### P\_FRAME# 3-1, 3-5, 5-10, 13-2

### P\_GNT# 3-1, 3-5, 5-10, 14-1

### P\_IDSEL 3-1, 3-5, 5-10, 8-8, 16-2

### P\_INTA# 3-1, 3-26, 5-6, 5-10, 6-110, 6-114, 15-2

### P\_IRDY# 3-1, 3-5, 5-10, 6-26, 6-92

### P\_LOCK# 3-1, 3-6, 5-10, 13-1–13-2

### P\_M66EN 3-1, 3-6, 4-1, 4-2, 5-2, 5-10

### P\_PAR 3-1, 3-6, 5-10, 14-1

### P\_PAR64 3-1, 3-6, 5-10, 14-1

### P\_PERR# 3-1, 3-6, 5-10, 12-1–12-13

### P\_PLEN# 3-2, 3-13, 4-3, 5-10, 25-3

### P\_PME# 3-21, 5-10, 6-115, 6-116, 21-1

### P\_REQ# 3-1, 3-6, 5-10, 14-1

### P\_REQ64# 3-1, 3-6, 5-10

### P\_RSTIN#

- 3-16, 5-6, 5-8, 5-9–5-13, 6-81, 6-98

**P\_RSTOUT#**

3-16, 5-2, 5-6, 5-7, 5-8, 5-11, 6-128

**P\_SERR#** 3-1, 3-7, 5-11, 6-2, 6-5, 6-15, 6-32, 6-34,  
6-63, 6-83, 6-103, 8-4, 8-7, 8-13, 8-14, 8-15, 8-16,  
12-1–12-13, 13-2

**P\_STOP#** 3-1, 3-7, 5-11, 12-3

**P\_TRDY#** 3-1, 3-7, 5-11, 12-2, 12-3

**P\_TST[1:0]** 3-21, 5-11

**P\_XCAP** 3-22, 5-1, 5-11, 22-2

package specs 26-1–26-4

parity 12-1–12-13

error rules 9-3

primary signal 3-6

reporting errors 17-2

secondary signal 3-10, 3-11

**PBGA**

industry standard 26-1

package

ordering information D-1

pinout 26-4–26-5

**PCI 6540**

general product information 1-1–1-5, D-1

*PCI 6520BB and PCI 6540BB Pin Comparison,  
Application Note C-1*

PCI arbitration 14-1–14-3

PCI Bus operation 8-1–8-18

*PCI Bus Power Management Interface Specification,  
Revision 1.1*

*See PCI Power Mgmt. r1.1*

PCI Configuration registers 6-2–6-136

*PCI Local Bus Specification, Revision 2.1*

*See PCI r2.1*

*PCI Local Bus Specification, Revision 2.3*

*See PCI r2.3*

*PCI Power Mgmt. r1.1* 2-1, 21-1

*PCI r2.1* 6-43, 6-44, 6-45

*PCI r2.3* 1-4, 3-1, 3-2, 3-3, 5-2, 6-43, 6-44, 6-45, 10-5,  
11-1, 11-4, 23-1, 24-2

PCI Shadow Configuration registers 6-81–6-136

*PCI to PCI Bridge Architecture Specification, Revision  
1.1*

*See P-to-P Bridge r1.1*

PCI transactions 8-1–8-18, 11-1–11-4

PCI Type 0 Header registers

primary port 6-62–6-71

secondary port 6-72–6-80

PCI Type 1 Header registers 6-4–6-15

**PCIBAR0** register 2-2, 6-43, 6-44, 6-67, 10-9, 10-10,  
10-11

**PCIBAR1** register 2-2, 6-43, 6-48, 6-68, 6-124, 10-9,  
10-10, 10-11

**PCIBAR2** register 2-2, 6-43, 6-69, 10-9, 10-10, 10-11

**PCIBISTR** register 6-7, 6-66, 6-75, 7-3

**PCICCR** register 6-6, 6-37, 6-65, 6-74, 6-107

**PCICLSR** register 6-6, 6-22, 6-23, 6-66, 6-89, 8-3, 18-1

**PCICR** register 6-4–6-5, 6-6, 6-10, 6-15, 6-32, 6-63,  
6-64, 6-73, 6-74, 6-82, 6-103, 6-104, 7-3, 8-4, 8-7,  
8-13, 8-14, 8-15, 8-16, 8-18, 10-1, 10-2, 10-4, 12-1,  
12-2, 12-3, 12-5, 12-6, 12-7–12-13, 13-2, 17-2

**PCIHTR** register 6-7, 6-37, 6-66, 6-75, 6-107

**PCIIDR** register 6-4, 6-37, 6-62, 6-72, 6-107, 7-3

**PCIOBAR** register 6-9, 10-1, 10-2, 10-6

**PCIOBARU16** register 6-9, 6-13, 10-2

**PCIIOLMT** register 6-9, 10-2, 10-6

**PCIIOLMTU16** register 6-9, 6-13, 10-2

**PCIIPR** register 6-14

**PCILTR** register 6-6, 6-66

**PCIMBAR** register 6-11, 10-2, 10-3, 10-6

**PCIMLMT** register 6-11, 10-2, 10-3, 10-6

**PCIPBNO** register 6-8, 6-135, 8-11

**PCIPILR** register 6-71

**PCIPIPR** register 6-71

**PCIPMBAR** register 6-12, 6-13, 10-2, 10-4

**PCIPMBARU32** register 6-12, 6-13, 10-2

**PCIPMGR** register 6-71, 6-107

**PCIPMLMT** register 6-12, 6-13, 10-2, 10-4

**PCIPMLMTU32** register 6-12, 6-13, 10-2

**PCIPMLR** register 6-71, 6-107

**PCIREV** register 6-6, 6-65, 6-74

**PCISBNO** register 6-8, 8-9, 8-11

**PCISCLSR** register 6-75, 8-3, 18-1

**PCISCR** register 6-72, 6-104

**PCISID** register 6-70, 6-79, 6-107, 7-4

**PCISILR** register 6-80

**PCISIPR** register 6-80

**PCISLTR** register 6-8, 6-75

**PCISMGR** register 6-80, 6-107

**PCISMLR** register 6-80, 6-107

**PCISR** register 6-5, 6-64, 8-13, 8-14, 8-15, 8-16, 8-18,  
12-1, 12-2, 12-3, 12-4, 12-6, 12-7, 12-9, 17-1

**PCISSR** register 6-10, 6-73, 8-13, 8-14, 8-15, 8-16,  
8-18, 12-1, 12-2, 12-3, 12-4, 12-6, 12-8, 12-10, 17-1

**PCISUBNO** register 6-8, 8-11

**PCISVID** register 6-70, 6-79, 6-107, 7-4

**PCIUBAR0** register 6-44, 6-76, 10-10

**PCIUBAR1** register 6-44, 6-46, 6-77, 6-121, 10-10

**PCIUBAR2** register 6-45, 6-78, 10-10

*PCI-X Addendum to PCI Local Bus Specification,  
Revision 1.0b*

*See PCI-X r1.0b*

PCI-X Bus operation

9-1–9-21

- PCI-X Capability registers
  - 6-56–6-59, 6-134–6-136
- PCI-X clock B-1–B-2
- PCI-X r1.0b* 3-1, 3-2, 6-1, 11-1, 11-4, 25-4
- PCI-X transactions 9-1–9-21, 11-4–11-6
- PCIX\_CMD register 6-134
- PCIX\_NEXT register 6-56, 6-134
- PCIXBSR register 3-21, 6-58, 6-135, 9-7, 9-11, 9-13
- PCIXCAPID register 6-56, 6-134
- PCIXDNSTR register 6-57, 6-59, 6-136, 19-2
- PCIXSSR register 3-20, 6-56–6-57, 15-2
- PCIXUPSTR register 6-58, 6-59, 6-136, 19-4
- PFTCR register 6-18, 6-84, 7-3, 18-1, 18-2
- physical specs 26-1–26-4
- PICMG 2.1 R2.0* 2-1, 22-1
- PICMG 2.1 R2.0 Hot Swap Specification*
  - See *PICMG 2.1 R2.0*
- PINPCNT register 6-22, 6-23, 6-88, 6-89, 7-3, 18-1, 18-3, 19-2, 19-4
- pinout
  - PBGA 26-4–26-5
  - specs 26-1–26-4
- pins
  - Boundary Scan 3-2, 3-18
  - BPCC\_EN 3-21, 5-9
  - Clock Related 3-2, 3-13–3-15
  - CompactPCI Hot Swap 3-2, 3-17
  - DEV64# 3-21, 5-9, 6-29, 6-58, 6-95, 6-135
  - EEPCLK 3-18, 5-9, 7-1
  - EEPDATA 3-18, 5-9, 7-1
  - EJECT 3-17, 5-9, 22-1
  - ENUM# 3-17, 5-9, 6-54, 6-132, 22-1, 22-2, A-5
  - GPIO 3-3, 3-19–3-20, 22-2
  - GPIO[15:14, 12:8] 3-19
  - GPIO[3:0] 3-19
  - GPIO[7:4] 3-19
  - GPIO13 pin 3-20, 5-1, 15-2
  - Hot Swap 3-17
  - JTAG 3-2, 3-18, 24-1
  - L\_STAT 3-17, 5-9, 22-1
  - Miscellaneous 3-3, 3-21–3-23
  - MSK\_IN 3-2, 3-13, 5-9
  - Multiplexed 3-1, 3-3, 3-25–3-26
  - NC 3-24
  - No Connect 3-1, 3-24, C-1
  - OSCIN 3-2, 3-13, 4-1, 4-2, 5-2, 5-9
  - OSCSSEL# 3-2, 3-13, 4-1, 4-2, 5-9
  - P\_ACK64# 3-1, 3-4, 5-9
  - P\_AD[31:0] 3-4, 16-2
  - P\_AD[63:0] 3-1
  - P\_AV<sub>DD</sub> 3-24, 5-9
  - P\_AV<sub>SS</sub> 3-24, 5-9
  - P\_BOOT 3-21, 5-6, 5-7, 5-10, 21-1
  - P\_CBE[3:0]# 3-5, 5-10, 8-2, 8-9, 16-1–16-2
  - P\_CBE[7:0]# 3-1
  - P\_CBE[7:4]# 3-5, 5-10, 8-2, 8-9
  - P\_CLKIN 3-2, 3-13, 4-1, 4-4, 5-2, 5-6, 5-7, 5-9, 5-10, 22-1
  - P\_CLKOE 3-2, 3-13, 4-1, 5-10, 6-29, 6-95
  - P\_CLKRUN# 3-26, 5-10
  - P\_CR 3-2, 3-13, 5-10
  - P\_DEVSEL# 3-1, 3-5, 5-10, 8-8, 12-1
  - P\_FRAME# 3-1, 3-5, 5-10, 13-2
  - P\_GNT# 3-1, 3-5, 5-10, 14-1
  - P\_IDSEL 3-1, 3-5, 5-10, 8-8, 16-2
  - P\_INTA# 3-1, 3-26, 5-6, 5-10, 6-110, 6-114, 15-2
  - P\_IRDY# 3-1, 3-5, 5-10, 6-26, 6-92
  - P\_LOCK# 3-1, 3-6, 5-10, 13-1–13-2
  - P\_M66EN 3-1, 3-6, 4-1, 4-2, 5-2, 5-10
  - P\_PAR 3-1, 3-6, 5-10, 14-1
  - P\_PAR64 3-1, 3-6, 5-10, 14-1
  - P\_PERR# 3-1, 3-6, 5-10, 12-1–12-13
  - P\_PLEN# 3-2, 3-13, 4-3, 5-10, 25-3
  - P\_PME# 3-21, 5-10, 6-115, 6-116, 21-1
  - P\_REQ# 3-1, 3-6, 5-10, 14-1
  - P\_REQ64# 3-1, 3-6, 5-10
  - P\_RSTIN# 3-16, 3-19, 5-6, 5-8, 5-9–5-13, 6-81, 6-98, 15-2
  - P\_RSTOUT# 3-16, 5-2, 5-6, 5-7, 5-8, 5-11, 6-128
  - P\_SERR# 3-1, 3-7, 5-11, 6-2, 6-5, 6-15, 6-32, 6-34, 6-63, 6-83, 6-103, 8-4, 8-7, 8-13, 8-14, 8-15, 8-16, 12-1–12-13, 13-2
  - P\_STOP# 3-1, 3-7, 5-11, 12-3
  - P\_TRDY# 3-1, 3-7, 5-11, 12-2, 12-3
  - P\_TST[1:0] 3-21, 5-11
  - P\_XCAP 3-22, 5-1, 5-11, 22-2
  - Power and Ground 3-24, 4-3, 25-3
  - Primary Clock 3-2
  - Primary PCI Bus Interface 3-1–3-2, 3-4–3-7
  - PRV\_DEV 3-25, 5-11, 6-16, 6-36, 6-40, 6-118, 7-2, 10-5, 22-2, A-2
  - PWRGD 3-16, 5-2, 5-4–5-5, 5-6, 5-7, 5-8, 5-9–5-13, 5-14, 15-2
  - REFCLK 3-2, 3-13, 4-4, 5-11
  - ReserveIO[2:1] 3-22
  - Reset 3-2, 3-16–3-17, 5-9
  - S\_ACK64# 3-1, 3-8
  - S\_AD[31:0] 3-8, 16-4
  - S\_AD[63:0] 3-1
  - S\_AD[63:32] 3-8, 6-37, 6-107
  - S\_AV<sub>DD</sub> 3-24, 5-11
  - S\_AV<sub>SS</sub> 3-24, 5-11
  - S\_CBE[3:0]# 3-8, 16-3–16-4
  - S\_CBE[7:0]# 3-1
  - S\_CBE[7:4]# 3-9, 6-37, 6-107
  - S\_CFN# 3-22, 6-29, 6-95

S\_CLKIN 3-2, 3-14, 4-1, 4-2, 4-4, 6-29, 6-95, A-2, A-3, A-5, A-6  
S\_CLKIN\_STB 3-2, 3-14  
S\_CLKO[4:1] 3-2, 3-15, 4-1, 5-11  
S\_CLKO0 3-2, 3-14, 3-15, 4-1, 4-2, 5-11, 6-34, 6-102, A-5, A-6, B-2  
S\_CLKOFF 3-2, 3-15, 4-1, 4-2  
S\_CLKRUN# 3-26, 5-12  
S\_CR 3-2, 3-15, 5-12  
S\_DEVSEL# 3-1, 3-9, 5-6, 5-7, 5-12, 12-2  
S\_FRAME# 3-1, 3-9, 5-12, 13-2, 14-2, 14-3  
S\_GNT[7:0]# 3-1  
S\_GNT[7:1]# 3-9, 5-12  
S\_GNT0# 3-9, 5-12, 14-3, A-4, A-5, A-7  
S\_IDSEL 3-1, 3-10, 5-12, 10-5  
S\_INTA# 3-1, 3-26, 5-6, 5-12, 6-110, 6-114, 15-2  
S\_IRDY# 3-1, 3-10, 5-12, 6-18, 6-84, 14-3  
S\_LOCK# 3-1, 3-10, 5-12, 13-1–13-2  
S\_M66EN 3-1, 3-10, 4-1, 4-2, 5-1, 5-2, 5-12  
S\_PAR 3-1, 3-10, 5-12, 14-3  
S\_PAR64 3-1, 3-10, 5-12, 6-37, 6-107, 14-3  
S\_PERR# 3-1, 3-11, 5-12, 6-10, 6-73, 12-1–12-13  
S\_PLEN# 3-15, 4-3, 5-12, 25-3  
S\_PME# 3-22, 5-12, 6-115, 6-116, 21-1  
S\_REQ[7:0]# 3-1  
S\_REQ[7:1]# 3-11, 5-12  
S\_REQ0# 3-11, 5-12, 14-3, A-4, A-5, A-6, A-7  
S\_REQ64# 3-1, 3-11, 5-12  
S\_RSTIN# 3-16, 5-2, 5-4–5-5, 5-6, 5-7, 5-8, 5-9–5-13, 6-41, 6-81, 6-98, 6-115, 6-116, 6-119, 6-123, 6-128, A-6  
S\_RSTOUT# 3-17, 4-1, 5-2, 5-5, 5-6, 5-7, 5-8, 5-12, 6-82, 6-128, 21-1, A-3, A-5  
S\_SERR# 3-1, 3-12, 12-1–12-13  
S\_STOP# 3-1, 3-12, 5-6, 5-7, 5-13, 12-3  
S\_TRDY# 3-1, 3-12, 5-6, 5-7, 5-13, 12-3  
S\_TST[1:0] 3-23, 5-13  
S\_XCAP\_IN 3-23, 5-1, 5-13  
S\_XCAP\_PU 3-23, 5-1, 5-13  
Secondary Clock 3-2  
Secondary PCI Bus Interface 3-1–3-2, 3-8–3-12  
Serial EEPROM 3-3, 3-18  
TCK 3-18, 24-1  
TDI 3-18, 24-1, 24-2  
TDO 3-18, 24-1, 24-2  
TEST 3-23, 5-13  
TMS 3-18, 24-1  
TRANS# 3-23, 4-2, 5-6, 5-7, 6-29, 6-60, 6-95, 20-1, A-5  
TRDY# 24-1  
TRST# 3-18, 24-2  
U\_MODE 3-23, 4-2, 5-6, 5-7, 6-29, 6-95, A-4, A-5  
V<sub>DD\_CORE</sub> 3-24, 5-13  
V<sub>DD\_IO</sub> 3-24, 5-13  
V<sub>SS</sub> 3-13, 3-15, 3-24, 5-13

XB\_MEM 3-25, 5-13, 6-67, 6-68, 6-69, 6-76, 6-77, 6-78, 6-123, 6-126, 20-1–20-2, A-3, A-4, A-7  
*See Also* pinout designations and Appendix C, *PCI 6520BB and PCI 6540BB Pin Comparison*  
PITLPCNT register  
6-22, 6-23, 6-88, 6-89, 7-3, 18-1, 18-2, 19-2, 19-4  
PLX Technology, Inc.  
product information 1-1  
product ordering and technical support D-2  
PMAXPCNT register 6-25, 6-27, 6-89, 6-91, 7-3, 18-1, 18-3, 19-3  
PMC register 6-37, 6-51, 6-52, 6-107, 6-130, 7-4  
PMCAPID register 6-51, 6-129  
PMCDATA register 6-37, 6-53, 6-107, 6-131, 7-3  
PMCSR register 5-7, 6-3, 6-37, 6-53, 6-99, 6-107, 6-131, 7-4  
PMCSR\_BSE register 6-53, 6-131  
PMNEXT register 6-51, 6-129  
Power and Ground pins 3-24, 4-3, 25-3  
power good  
input signal 3-16  
reset 5-2–5-5  
*See Also* PWRGD  
Power Management 21-1  
Power Management Capability registers 6-3, 6-51–6-53, 6-99, 6-107, 6-129–6-131, 7-4  
power supply 3-3  
Power-Up Status registers 3-19, 3-20, 6-3, 6-39, 6-98, 6-108–6-109  
prefetch 18-3  
data timeout flushing 19-3  
incremental count 6-2, 6-23, 6-24, 6-81, 6-89, 6-90, 7-3  
memory 6-12–6-13, 10-3  
read transaction 8-5–8-6  
reprogramming registers 18-1  
setting byte count 19-4  
smart 6-27, 19-3  
Prefetch Control registers 6-2, 6-22–6-25, 6-88–6-91, 7-3, 18-1, 18-2, 19-2, 19-4  
Primary  
clock frequency measurement 4-4  
Clock pins 3-2  
Configuration registers 5-6, 6-62–6-80  
Flow-Through Control registers 6-18, 6-82–6-84  
PCI Bus Interface pins 3-1–3-2, 3-4–3-7  
priority schemes 14-2–14-3  
private memory 6-2, 6-16, 6-36, 10-5, A-2  
PRV\_DEV 3-25, 5-11, 6-16, 6-36, 6-40, 6-118, 7-2, 10-5, 22-2, A-2  
PSERRED register 6-32, 6-103, 8-13–8-16, 12-1, 12-4  
PSERRSR register 6-34, 12-1  
PSSERRSR register  
6-105, 12-1, 12-4

## *P-to-P Bridge r1.1*

6-1, 10-3, 12-6  
pull-up/pull-down resistor recommendations 3-1–3-3  
PVPD\_NEXT register 6-55, 6-133  
PVPDAD register 6-55, 6-133  
PVPDATA register 6-55, 6-133  
PVPDID register 6-55, 6-133  
PVTMBAR register 3-25, 6-16, 6-36  
PVTMBARU32 register 3-25, 6-36  
PVTMLMT register 3-25, 6-36  
PVTMLMTU32 register 6-36  
PWRGD 3-16, 5-2, 5-4–5-5, 5-6, 5-7, 5-8, 5-9–5-13,  
5-14, 15-2  
PWRUPSR register 3-19, 3-20, 6-39, 6-109

## **R**

Read-Only Control register 6-107  
REFCLK 3-2, 3-13, 4-4, 5-11  
registers

ACNTRL 6-17, 6-28, 6-128, 14-1, 14-2  
Address Translation Control 6-42–6-50, 6-120–6-126,  
10-5, 10-11, 20-1  
Arbiter Control 6-2, 6-16–6-17, 6-99, 6-127–6-128, 14-1  
BCNTRL 5-8, 6-14–6-15, 6-17, 6-19, 6-82, 6-128, 19-3  
BUFCR 6-27, 6-93, 19-3, 19-4  
Buffer Control 6-2, 6-27, 6-81, 6-93  
CAP\_PTR 6-5, 6-14, 6-64, 6-70, 6-73, 6-79  
CCNTRL 3-25, 6-16, 6-36, 6-81, 6-117, 6-127, 8-5, 10-5  
Chip Control 3-25, 6-2, 6-16–6-17, 6-36, 6-81, 6-99,  
6-127–6-128, 8-5  
CLKCNTRL 3-2, 3-13, 3-15, 4-1, 5-6, 6-34, 6-102  
CLKRUN 6-35  
Control 2-1, 6-16–6-17, 6-127–6-128, 8-6  
DCNTRL 3-16, 5-7, 5-8, 6-17, 6-128  
Device-Specific 6-16–6-17  
Diagnostic Control 3-16, 5-6, 5-7, 6-16–6-17, 6-127–  
6-128  
Direct Message Interrupt 5-6, 6-110–6-111, 20-2  
Doorbell Interrupt 6-99, 6-114–6-116, 20-1, 20-2  
DWNBAR0MSK 6-49, 6-67, 6-125, 7-4, 10-9, 10-11  
DWNBAR1MSK 6-43, 6-48, 6-49, 6-68, 6-69, 6-124,  
6-125, 7-4, 10-9, 10-11  
DWNBAR2MSK 6-50, 6-69, 6-125, 7-4, 10-9, 10-11  
DWNDBIE 6-114  
DWNDBIR 6-114  
DWNDBIS 6-115  
DWNINTE 5-6, 6-116  
DWNINTSR 5-6, 6-111, 6-115  
DWNMSG0 6-111  
DWNMSG1 6-111  
DWNMSG2 6-111  
DWNMSG3 6-111  
DWNTNBAR0 6-48, 6-124, 7-4, 10-12  
DWNTNBAR1 6-48, 6-124, 10-9, 10-12

DWNTNBAR2 6-48, 6-124, 10-9, 10-12  
DWNTNE 6-50, 6-126, 7-4, 10-10, 10-12, 20-1  
EEPADDR 6-30, 6-96, 7-1  
EEPCNTRL 6-29, 6-95, 7-1  
EEPDATA 6-30, 6-96, 7-1  
Extended 5-2, 5-14, 6-3, 6-40, 6-45, 6-99, 6-118, 6-119–  
6-120, 10-6, 10-7  
EXTRDATA 5-14, 6-40, 6-41, 6-67–6-69, 6-76–6-78,  
6-119, 6-120, 10-6  
EXTRIDX 5-14, 6-41, 6-67–6-69, 6-76–6-78, 6-119, 10-6  
GPIOID[15:8] 6-39, 6-109  
GPIOID[3:0] 6-33, 6-106, 15-1  
GPIOID[7:4] 6-38, 6-108, 15-1  
GPIOOD[15:8] 6-39, 6-109  
GPIOOD[3:0] 6-33, 6-106, 15-1  
GPIOOD[7:4] 6-38, 6-108, 15-1  
GPIOOE[15:8] 3-19, 3-20, 6-39, 6-109  
GPIOOE[3:0] 3-19, 6-33, 6-106, 15-1  
GPIOOE[7:4] 3-19, 6-38, 6-108, 15-1  
Header 6-4–6-15, 6-62–6-71, 6-72–6-80  
Hot Swap 3-17, 6-37, 6-54, 6-107, 6-132, 22-1, 22-2  
HS\_CNTL 3-17, 6-54, 6-132  
HS\_CSR 3-17, 6-54, 6-132, 22-1, 22-2  
HS\_NEXT 3-17, 6-54, 6-132  
HSSRRC 3-17, 6-3, 6-37, 6-51, 6-107, 10-6, 10-9, 10-10,  
14-3  
IACNTRL 6-28, 6-94, 7-3, 14-1, 14-2, 14-3  
Internal Arbiter Control 6-2, 6-28, 6-81, 6-93, 6-94, 7-3,  
14-1, 14-2  
Interrupt 6-110–6-116, 20-2  
Message Signaled Interrupt 6-112–6-113, 20-2  
Miscellaneous Options 6-2, 6-20–6-21, 6-81, 7-3, 8-3,  
17-3  
MSCOPT 6-20–6-21, 6-86–6-87, 7-3, 8-3, 8-9, 11-2,  
13-2, 17-3  
MSIADDR 6-113  
MSIC 6-112  
MSICAPID 6-107, 6-112  
MSIDATA 6-113  
MSINEXT 6-112  
MSIUADDR 6-113  
Non-Transparent Configuration Ownership  
Semaphore 6-62, 6-98, 6-99, 6-117  
Non-Transparent mode 6-60–6-136  
NTCOS 6-62, 6-117  
PCI Configuration 6-2–6-136  
PCI Shadow Configuration 6-81–6-136  
PCI Type 1 Header 6-4–6-15  
PCIBAR0 2-2, 6-43, 6-44, 6-67, 10-9, 10-10, 10-11  
PCIBAR1 2-2, 6-43, 6-48, 6-68, 6-124, 10-9, 10-10,  
10-11  
PCIBAR2 2-2, 6-43, 6-69, 10-9, 10-10, 10-11  
PCIBISTR 6-7, 6-66, 6-75, 7-3  
PICCCR 6-6, 6-37, 6-65, 6-74, 6-107  
PICLSR 6-6, 6-22, 6-23, 6-66, 6-89, 8-3, 18-1

PCICR 6-4–6-5, 6-6, 6-10, 6-15, 6-32, 6-63, 6-64, 6-73, 6-74, 6-82, 6-103, 6-104, 7-3, 8-4, 8-7, 8-13, 8-14, 8-15, 8-16, 8-18, 10-1, 10-2, 10-4, 12-1, 12-2, 12-3, 12-5, 12-6, 12-7–12-13, 13-2, 17-2  
PCIHTR 6-7, 6-37, 6-66, 6-75, 6-107  
PCIIDR 6-4, 6-37, 6-62, 6-72, 6-107, 7-3  
PCIIOBAR 6-9, 10-1, 10-2, 10-6  
PCIIOBARU16 6-9, 6-13, 10-2  
PCIIOLMT 6-9, 10-2, 10-6  
PCIIOLMTU16 6-9, 6-13, 10-2  
PCIIPR 6-14  
PCILTR 6-6, 6-66  
PCIMBAR 6-11, 10-2, 10-3, 10-6  
PCIMLMT 6-11, 10-2, 10-3, 10-6  
PCIPBNO 6-8, 8-11  
PCIPILR 6-71  
PCIPIPR 6-71  
PCIPMBAR 6-12, 6-13, 10-2, 10-4  
PCIPMBARU32 6-12, 6-13, 10-2  
PCIPMBNO 6-135  
PCIPMGR 6-71, 6-107  
PCIPMLMT 6-12, 6-13, 10-2, 10-4  
PCIPMLMTU32 6-12, 6-13, 10-2  
PCIPMLR 6-71, 6-107  
PCIREV 6-6, 6-65, 6-74  
PCISBNO 6-8, 8-9, 8-11, 9-11  
PCISCLSR 6-75, 8-3, 18-1  
PCISCR 6-72, 6-104  
PCISID 6-70, 6-79, 6-107, 7-4  
PCISILR 6-80  
PCISIPR 6-80  
PCISLTR 6-8, 6-75  
PCISMGR 6-80, 6-107  
PCISMLR 6-80, 6-107  
PCISR 6-5, 6-64, 8-13, 8-14, 8-15, 8-16, 8-18, 12-1, 12-2, 12-3, 12-4, 12-6, 12-7, 12-9, 17-1  
PCISRR 12-4  
PCISSR 6-10, 6-73, 8-13, 8-14, 8-15, 8-16, 8-18, 12-1, 12-2, 12-3, 12-4, 12-6, 12-8, 12-10, 17-1  
PCISUBNO 6-8, 8-11  
PCISVID 6-70, 6-79, 6-107, 7-4  
PCIUBAR0 6-44, 6-76, 10-10  
PCIUBAR1 6-44, 6-46, 6-77, 6-121, 10-10  
PCIUBAR2 6-45, 6-78, 10-10  
PCIX\_CMD 6-134  
PCIX\_NEXT 6-56, 6-134  
PCIXBSR 3-21, 6-58, 6-135, 9-7, 9-11, 9-13  
PCIXCAPID 6-56, 6-134  
PCIXDNSTR 6-57, 6-59, 6-136, 19-2  
PCIXSSR 3-20, 6-56–6-57, 15-2  
PCIXUPSTR 6-58, 6-59, 6-136, 19-4  
PFTCR 6-18, 6-84, 7-3, 18-1, 18-2  
PINPCNT 6-22, 6-23, 6-88, 6-89, 7-3, 18-1, 18-3, 19-2, 19-4  
PITLPCNT 6-22, 6-23, 6-88, 6-89, 7-3, 18-1, 18-2, 19-2, 19-4  
PMAXPcnt 6-25, 6-27, 6-89, 6-91, 7-3, 18-1, 18-3, 19-3  
PMC 6-37, 6-51, 6-52, 6-107, 6-130, 7-4  
PMCAPID 6-51, 6-129  
PMCDATA 6-37, 6-53, 6-107, 6-131, 7-3  
PMCSR 5-7, 6-3, 6-37, 6-53, 6-99, 6-107, 6-131, 7-4  
PMCSR\_BSE 6-53, 6-131  
PMNEXT 6-51, 6-129  
Power Management Capability 6-3, 6-51–6-53, 6-99, 6-107, 6-129–6-131, 7-4  
Power-Up Status 3-19, 3-20, 6-3, 6-39, 6-98, 6-108–6-109  
Prefetch Control 6-22–6-25, 6-88–6-91, 7-3, 18-1, 18-2, 19-2, 19-4  
Primary Configuration 5-6, 6-62–6-80  
Primary Flow-Through Control 6-18, 6-82–6-84  
PSERRED 6-32, 6-103, 8-13–8-16, 12-1, 12-4  
PSERRSR 6-34, 12-1  
PSSERRSR 6-105, 12-1, 12-4  
PVPD\_NEXT 6-55, 6-133  
PVPDAD 6-55, 6-133  
PVPDATA 6-55, 6-133  
PVPDID 6-55, 6-133  
PVTMBAR 3-25, 6-16, 6-36  
PVTMBARU32 3-25, 6-36  
PVTMLMT 6-36  
PVTMLMTU32 6-36  
PVTMMLMT 3-25  
PWRUPSR 3-19, 3-20, 6-39, 6-109  
Read-Only Control 6-107  
SCRATCHx 5-6, 5-8, 5-14, 6-40, 6-41, 6-118, 6-119  
Secondary Flow-Through Control 6-2, 6-26, 6-81, 6-92, 7-3  
Semaphore 6-62, 6-98, 6-117  
Serial EEPROM 6-29–6-30, 6-95–6-96, 7-2  
SFTCR 6-26, 6-92, 7-3, 18-1, 18-2  
SINPCNT 6-23, 6-24, 6-89, 6-90, 7-3, 18-1, 18-3, 19-2, 19-4  
SITLPCNT 19-2  
SITLPCNT 6-23, 6-24, 6-89, 6-90, 7-3, 18-1, 18-2, 19-4  
SMAXPCNT 6-24, 6-25, 6-27, 6-90, 6-91, 7-3, 18-1, 18-3, 19-3  
SSERRED 6-104, 12-1  
Sticky Scratch 5-6, 5-8, 5-14, 6-40, 6-41, 6-118, 6-119  
System Error Event 6-32, 6-103–6-105  
TEST 6-29, 6-95  
Timeout Control 6-2, 6-19, 6-81, 6-85, 7-3, 8-4  
Timer 4-4, 6-31, 6-85, 6-97  
TMRCNT 4-4, 6-31, 6-97  
TMRCNTRL 4-4, 6-31, 6-97  
TOCNTRL 6-19, 6-32, 6-85, 6-103, 6-104, 7-3, 8-4  
Transparent mode 6-2–6-42  
UPSBAR0MSK 6-47, 6-76, 6-122, 7-4, 10-10

UPSBAR1MSK 6-45, 6-46, 6-47, 6-77, 6-78, 6-121, 6-122, 7-4, 10-10  
 UPSBAR2MSK 6-47, 6-78, 6-122, 7-4, 10-10  
 UPSDBIE 6-114  
 UPSDBIR 6-114  
 UPSDBIS 6-116  
 UPSINTE 6-115  
 UPSINTSR 5-6, 6-116  
 UPSMSG0 5-6, 6-110  
 UPSMSG1 5-6, 6-110  
 UPSMSG2 5-6, 6-110  
 UPSMSG3 5-6, 6-110  
 UPSTNBAR0 6-46, 6-121, 7-4, 10-10  
 UPSTNBAR1 6-46, 6-121, 7-4, 10-10  
 UPSTNBAR2 6-46, 6-121, 7-4  
 UPSTNE 6-48, 6-123, 7-4, 10-12, 20-1  
 VPD 6-55, 6-133, 23-1  
 XBCOS 6-101  
 XBDWNCA 6-100  
 XBDWNCD 6-100  
 XBDWNCOS 6-101  
 XBUPSCA 6-100  
 XBUPSCD 6-100  
 XBUPSCOS 6-101  
 ReserveIO[2:1]  
   3-22  
 reset 5-1–5-14  
   JTAG 24-1, 24-2  
   pins 3-2, 3-16–3-17  
 resistor recommendations, pull-up/pull-down 3-1–3-3  
 rotating priority scheme 14-2  
 RSTIN# 3-19, 15-2  
 rules 9-1–9-3, 11-1, 11-2, 11-5

## S

S\_ACK64# 3-1, 3-8  
 S\_AD[31:0] 3-8, 16-4  
 S\_AD[63:0] 3-1  
 S\_AD[63:32] 3-8, 6-37, 6-107  
 S\_AV<sub>DD</sub> 3-24, 5-11  
 S\_AV<sub>SS</sub> 3-24, 5-11  
 S\_CBE[3:0]# 3-8, 16-3–16-4  
 S\_CBE[7:0]# 3-1  
 S\_CBE[7:4]# 3-9, 6-37, 6-107  
 S\_CFN# 3-22, 6-29, 6-95  
 S\_CLKIN 3-2, 3-14, 4-1, 4-2, 4-4, 6-29, 6-95, A-2, A-3, A-5, A-6  
 S\_CLKIN\_STB 3-2, 3-14  
 S\_CLKO[4:1] 3-2, 3-15, 4-1, 5-11  
 S\_CLKO0 3-2, 3-14, 3-15, 4-1, 4-2, 5-11, 6-34, 6-102, A-5, A-6, B-2  
 S\_CLKOFF 3-2, 3-15, 4-1, 4-2  
 S\_CLKRUN# 3-26, 5-12

S\_CR 3-2, 3-15, 5-12  
 S\_DEVSEL# 3-1, 3-9, 5-6, 5-7, 5-12, 12-2  
 S\_FRAME# 3-1, 3-9, 5-12, 13-2, 14-2, 14-3  
 S\_GNT[7:0]# 3-1  
 S\_GNT[7:1]# 3-9, 5-12  
 S\_GNT0# 3-9, 5-12, 14-3, A-4, A-5, A-7  
 S\_IDSEL 3-1, 3-10, 5-12, 10-5  
 S\_INTA# 3-1, 3-26, 5-6, 5-12, 6-110, 6-114, 15-2  
 S\_IRDY# 3-1, 3-10, 5-12, 6-18, 6-84, 14-3  
 S\_LOCK# 3-1, 3-10, 5-12, 13-1–13-2  
 S\_M66EN 3-1, 3-10, 4-1, 4-2, 5-1, 5-2, 5-12  
 S\_PAR 3-1, 3-10, 5-12, 14-3  
 S\_PAR64 3-1, 3-10, 5-12, 6-37, 6-107, 14-3  
 S\_PERR# 3-1, 3-11, 5-12, 6-10, 6-73, 12-1–12-13  
 S\_PLLEN# 3-2, 3-15, 4-3, 5-12, 25-3  
 S\_PME# 3-22, 5-12, 6-115, 6-116, 21-1  
 S\_REQ[7:0]# 3-1  
 S\_REQ[7:1]# 3-11, 5-12  
 S\_REQ0# 3-11, 5-12, 14-3, A-4, A-5, A-6, A-7  
 S\_REQ64# 3-1, 3-11, 5-12  
 S\_RSTIN# 3-16, 5-2, 5-4–5-5, 5-6, 5-7, 5-8, 5-9–5-13, 6-41, 6-81, 6-98, 6-115, 6-116, 6-119, 6-123, 6-128, A-6  
 S\_RSTOUT# 3-17, 4-1, 5-2, 5-5, 5-6, 5-7, 5-8, 5-12, 6-82, 6-128, 21-1, A-3, A-5  
 S\_SERR# 3-1, 3-12, 12-1–12-13  
 S\_STOP# 3-1, 3-12, 5-6, 5-7, 5-13, 12-3  
 S\_TRDY# 3-1, 3-12, 5-6, 5-7, 5-13, 12-3  
 S\_TST[1:0] 3-23, 5-13  
 S\_XCAP\_IN 3-23, 5-1, 5-13  
 S\_XCAP\_PU 3-23, 5-1, 5-13  
 SAC 8-2, 9-2, 9-10, 10-3  
 SCRATCHx 5-6, 5-8, 5-14, 6-40, 6-41, 6-118, 6-119  
 Secondary  
   clock frequency measurement 4-4  
   Clock pins 3-2  
   Flow-Through Control register 6-2, 6-26, 6-81, 6-92, 7-3  
   PCI Bus Interface pins 3-1–3-2, 3-8–3-12  
 Semaphore registers 6-62, 6-98, 6-117  
 Serial EEPROM 6-2, 6-81, 7-1–7-4  
   pins 3-3, 3-18  
   registers 6-29–6-30, 6-95–6-96, 7-2  
 SFTCR register 6-26, 6-92, 7-3, 18-1, 18-2  
 signal specs 26-1–26-4  
 SINCPCNT register 6-23, 6-24, 6-89, 6-90, 7-3, 18-1, 18-3, 19-2, 19-4  
 Single Address Cycle  
   See SAC  
 SITLPCNT register 19-2  
 SITLPCNT register  
   6-23, 6-24, 6-89, 6-90, 7-3, 18-1, 18-2, 19-4

**smart prefetch**

6-27, 19-3

SMAXPCNT register 6-24, 6-25, 6-27, 6-90, 6-91, 7-3,  
18-1, 18-3, 19-3

**specs**

electrical 25-1  
mechanical 26-1–26-4

split completion 9-1–9-19

SSERRED register 6-104, 12-1

Sticky Scratch registers 5-6, 5-8, 5-14, 6-40, 6-41,  
6-118, 6-119

System Error Event registers 6-32, 6-103–6-105

**T**

**TAP controller**

*See* test access port controller

target abort 3-7, 3-12, 6-5, 6-10, 6-15, 6-32, 6-34, 6-64,  
6-73, 6-74, 6-82, 6-103, 6-104, 8-3, 8-4, 8-7, 8-12,  
8-13, 8-14, 8-15, 8-16, 8-18, 9-3, 9-4, 9-8, 9-9, 9-16,  
9-17, 9-18, 9-19, 12-1, 12-2, 12-4, 12-6, 13-2

TCK 3-18, 24-1

TDI 3-18, 24-1, 24-2

TDO 3-18, 24-1, 24-2

termination, abnormal 17-2

test access port controller 24-1, 24-2

TEST pin 3-23, 5-13

TEST register 6-29, 6-95

testability 24-1–24-2

timeout 19-3

control 6-2, 6-81, 7-3

Timeout Control register 6-2, 6-19, 6-81, 6-85, 7-3, 8-4

Timer registers 4-4, 6-31, 6-97

TMRCNT register 4-4, 6-31, 6-97

TMRCNTRL register 4-4, 6-31, 6-97

TMS 3-18, 24-1

TOCNTRL register 6-19, 6-32, 6-85, 6-103, 6-104, 7-3,  
8-4

TRANS# 3-23, 4-2, 5-6, 5-7, 6-29, 6-60, 6-95, 20-1, A-5

**transactions**

PCI 8-1–8-18, 11-1–11-4  
PCI-X 9-1–9-21, 11-4–11-6

**Transparent mode A-2**

address translation 10-6–10-10  
pin states 5-9–5-13  
pins 3-25–3-26  
registers 6-2–6-42  
reset inputs effect 5-8  
*See Also* Universal Transparent mode

TRDY# 24-1

TRST# 3-18, 24-2

**U**

U\_MODE 3-23, 4-2, 5-6, 5-7, 6-29, 6-95, A-4, A-5

Universal bridging A-4–A-5

**Universal Non-Transparent mode**

bus speed detection B-2  
groups 7-2  
reset inputs effect 5-8

**Universal Transparent mode**

bus speed detection B-2  
reset inputs effect 5-8

UPSBAR0MSK register 6-47, 6-76, 6-122, 7-4, 10-10

UPSBAR1MSK register 6-45, 6-46, 6-47, 6-77, 6-78,  
6-121, 6-122, 7-4, 10-10

UPSBAR2MSK register 6-47, 6-78, 6-122, 7-4, 10-10

UPSDDBIE register 6-114

UPSDDBIR register 6-114

UPSDDBIS register 6-116

UPSINTE register 6-115

UPSINTSR register 5-6, 6-116

UPSMMSG0 register 5-6, 6-110

UPSMMSG1 register 5-6, 6-110

UPSMMSG2 register 5-6, 6-110

UPSMMSG3 register 5-6, 6-110

UPSTNBAR0 register 6-46, 6-121, 7-4, 10-10

UPSTNBAR1 register 6-46, 6-121, 7-4, 10-10

UPSTNBAR2 register 6-46, 6-121, 7-4, 10-10

UPSTNE register 6-48, 6-123, 7-4, 10-12, 20-1

**V**

V<sub>DD\_CORE</sub> 3-24, 5-13

V<sub>DD\_IO</sub> 3-24, 5-13

VGA 6-14, 6-82, 10-1, 10-4

VHDL 24-2

VHSIC Hardware Description Language 24-2

VPD 23-1

registers 6-55, 6-133, 23-1

V<sub>SS</sub> 3-13, 3-15, 3-24, 5-13

**X**

XB\_MEM 3-25, 5-13, 6-67, 6-68, 6-69, 6-76, 6-77, 6-78,  
6-123, 6-126, 20-1–20-2, A-3, A-4, A-7

XBCOS register 6-101

XBDWNCA register 6-100

XBDWNCD register 6-100

XBDWNCOS register 6-101

XBUPSCA register 6-100

XBUPSCD register 6-100

XBUPSCOS register

6-101