



***Intel<sup>®</sup> 82801AA (ICH) and Intel<sup>®</sup>  
82801AB (ICH0) I/O Controller  
Hub***

**Datasheet**

---

***April, 1999***

Order Number: 290655-001





Information in this document is provided in connection with Intel products. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted by this document. Except as provided in Intel's Terms and Conditions of Sale for such products, Intel assumes no liability whatsoever, and Intel disclaims any express or implied warranty, relating to sale and/or use of Intel products including liability or warranties relating to fitness for a particular purpose, merchantability, or infringement of any patent, copyright or other intellectual property right. Intel products are not intended for use in medical, life saving, or life sustaining applications.

Intel may make changes to specifications and product descriptions at any time, without notice.

Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

The Intel® 82801AA (ICH) and Intel® 82801AB (ICH0) may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available upon request.

I<sup>2</sup>C is a two-wire communications bus/protocol developed by Philips. SMBus is a subset of the I<sup>2</sup>C bus/protocol and was developed by Intel. Implementations of the I<sup>2</sup>C bus/protocol or the SMBus bus/protocol may require licenses from various entities, including Philips Electronics N.V. and North American Philips Corporation.

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature may be obtained by:

calling 1-800-548-4725 or

by visiting Intel's website at <http://www.intel.com>.

Copyright © Intel Corporation, 1997-1998

\*Third-party brands and names are the property of their respective owners.



# Intel<sup>®</sup> 82801AA (I CH) and Intel<sup>®</sup> 82801AB (ICH0) Features

---

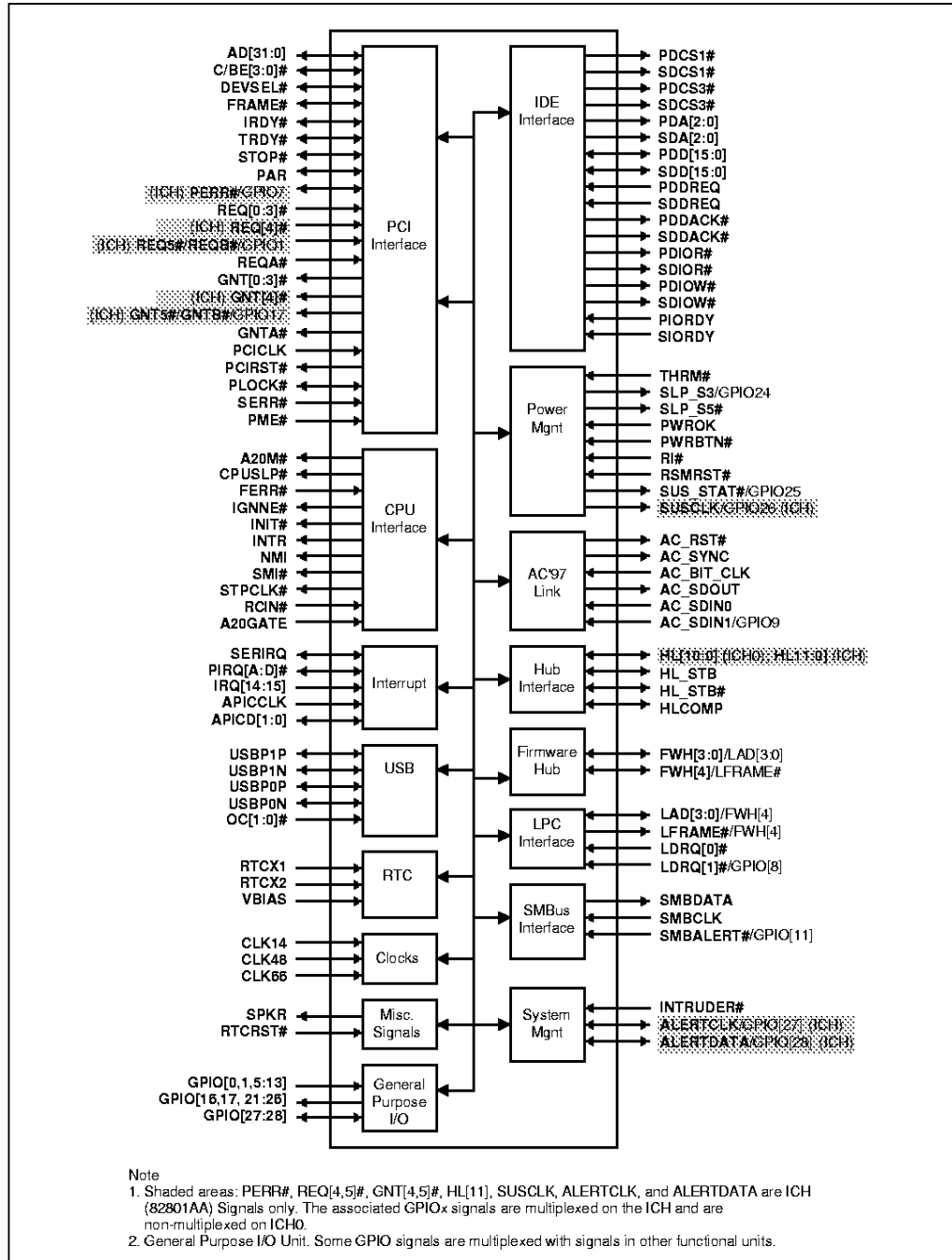
- PCI Bus Interface
  - Supports PCI at 33 MHz
  - Supports PCI Rev 2.2 Specification
  - 133 MByte/sec Maximum Throughput
  - Master PCI Device Support: up to 6 for the ICH and up to 4 for the ICH0
- Integrated IDE Controller
  - Independent Timing of Up to 4 Drives
  - The ICH Supports Ultra ATA/66 Mode (66 Mbytes/sec)
  - Supports Ultra ATA/33 Mode (33 Mbytes/sec)
  - PIO Mode 4 Transfers up to 14 Mbytes/s
  - Separate IDE Connections for Primary and Secondary Cables
  - Implements Write Ping-Pong Buffer for faster write performance
- USB
  - UHCI Implementation With 2 Ports
  - Supports Wake-up From Sleeping States S1-S4
  - Supports Legacy Keyboard/Mouse Software
  - USB Revision 1.1 Compliant
- AC'97 Link for Audio and Telephony CODECs
  - AC'97 2.1 Compliant
  - 5 Independent Bus Master Logic for PCM In, PCM Out, Mic Input, Modem In, Modem Out
  - Separate Independent PCI Functions for Audio and Modem
  - Supports wake-up events
- Interrupt Controller
  - Two Cascaded 82C59
  - Integrated I/O APIC Capability
  - 15 Interrupts Support in 8259 Mode, 24 Supported in I/O APIC Mode.
  - Supports Serial Interrupt Protocol
- Timers Based on 82C54
- System Timer, Refresh Request, Speaker Tone Output
- 3.3 V Operation With 5 V Tolerant Buffers for IDE and PCI signals.
- 241 BGA package
- GPIO
  - TTL, Open-Drain, Inversion
- Power Management Logic
  - ACPI 1.0 Compliant
  - Support for APM-Based Legacy Power Management for Non-ACPI Implementations
  - ACPI Defined Power States (S1, S3, S4, S5)
  - ACPI Power Management Timer
  - SMI# Generation
  - All Registers Readable/Restorable for Proper Resume from 0 V Suspend States
  - PCI PME#
- Low Pin count (LPC) Interface
  - Allows Connection of Legacy ISA and X-Bus Devices such as Super I/O
  - Supports Two Master/DMA Devices.
- Enhanced DMA Controller
  - Two Cascaded 8237 DMA Controllers
  - PCI DMA: Supports PC/PCI — Includes Two PC/PCI REQ#/GNT# Pairs
  - Supports LPC DMA
  - Supports DMA Collection Buffer to Provide Type-F DMA Performance for All DMA Channels
- Real - Time Clock
  - 256-byte Battery-Backed CMOS RAM
  - Hardware implementation to indicate Century Rollover
- System TCO Reduction Circuits
  - Timers to Generate SMI# and Reset Upon Detection of locked system
  - Timers to Detect Improper Processor Reset
  - Integrated Processor Frequency Strap Logic
- SM Bus
  - Host Interface Allows Processor to Communicate via SM Bus
  - Compatible With Most 2-Wire Components that are Also I<sup>2</sup>C compatible
- Supports ISA Bus via External PCI-ISA Bridge
- Firmware Hub (FWH) Interface
- The 82801AA provides Alert On LAN\* Support

This datasheet describes the Intel<sup>®</sup> 82801AA and Intel<sup>®</sup> 82801AB components. Non-shaded areas describe the functionality of both components.

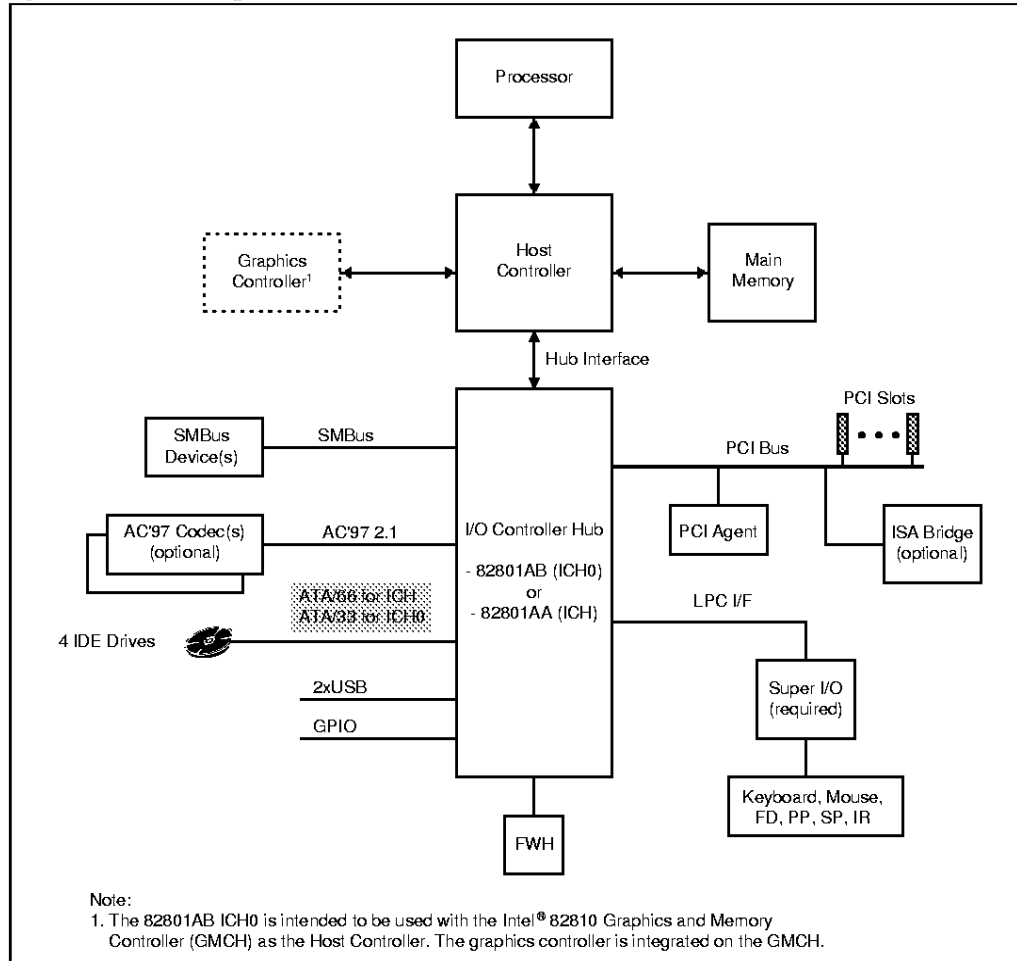
Shading, as is shown here, indicates differences between the two components.

The Intel<sup>®</sup> 82801AA and Intel<sup>®</sup> 82801AB may contain design defects or errors known as errata which may cause the products to deviate from published specifications. Current characterized errata are available on request.

### Intel® 82801AA (ICH) and Intel® 82801AB (ICH0) Simplified Block Diagram



### System Block Diagram



# Contents

---

|   |   |      |
|---|---|------|
| 1 | Introduction .....                                    | 1-1  |
|   | 1.1 About this Manual .....                           | 1-1  |
|   | 1.2 Overview .....                                    | 1-3  |
| 2 | Signal Description .....                              | 2-1  |
|   | 2.1 Hub Interface .....                               | 2-1  |
|   | 2.2 Firmware Hub Interface .....                      | 2-1  |
|   | 2.3 PCI Interface .....                               | 2-2  |
|   | 2.4 IDE Interface .....                               | 2-4  |
|   | 2.5 Low Pin Count (LPC) Interface .....               | 2-6  |
|   | 2.6 Interrupt Interface .....                         | 2-6  |
|   | 2.7 USB Interface .....                               | 2-6  |
|   | 2.8 Power Management Interface .....                  | 2-7  |
|   | 2.9 Processor Interface .....                         | 2-8  |
|   | 2.10 SMBus Interface .....                            | 2-9  |
|   | 2.11 System Management Interface .....                | 2-9  |
|   | 2.12 Real Time Clock Interface .....                  | 2-10 |
|   | 2.13 Other Clocks .....                               | 2-10 |
|   | 2.14 Miscellaneous Signals .....                      | 2-10 |
|   | 2.15 AC'97 Link .....                                 | 2-11 |
|   | 2.16 General Purpose I/O .....                        | 2-11 |
|   | 2.17 Power and Ground .....                           | 2-12 |
|   | 2.18 Pin Straps .....                                 | 2-13 |
|   | 2.18.1 Functional Strap .....                         | 2-13 |
|   | 2.18.2 Test Straps .....                              | 2-13 |
|   | 2.18.3 External RTC Circuitry .....                   | 2-14 |
|   | 2.18.4 5VREF / Vcc3_3 Sequencing Requirements .....   | 2-14 |
| 3 | Power Planes and Pin States .....                     | 3-1  |
|   | 3.1 Power Planes .....                                | 3-1  |
|   | 3.2 Output and I/O Signal Planes and States .....     | 3-1  |
|   | 3.3 Power Planes for Input Signals .....              | 3-4  |
|   | 3.4 Integrated Pull-Ups and Pull-Downs .....          | 3-5  |
|   | 3.5 IDE Integrated Series Termination Resistors ..... | 3-5  |
| 4 | Clock Domains .....                                   | 4-7  |
| 5 | Functional Description .....                          | 5-1  |
|   | 5.1 Hub Interface to PCI Bridge (D30:F0) .....        | 5-1  |
|   | 5.1.1 PCI Bus Interface .....                         | 5-1  |
|   | 5.1.2 PCI-to-PCI Bridge Model .....                   | 5-1  |
|   | 5.1.3 IDSEL to Device Number Mapping .....            | 5-1  |
|   | 5.1.4 SERR#/PERR#/NMI# Functionality .....            | 5-2  |



|          |  |      |
|----------|--|------|
| 5.2      | LPC Bridge (with System & Management Functions) (D31:F0) | 5-4  |
| 5.2.1    | LPC Interface  | 5-4  |
| 5.2.1.1  | LPC Cycle Types  | 5-4  |
| 5.2.1.2  | Start Field Definition                                   | 5-5  |
| 5.2.1.3  | Cycle Type / Direction (CYCTYPE + DIR)                   | 5-5  |
| 5.2.1.4  | SIZE   | 5-5  |
| 5.2.1.5  | SYNC   | 5-6  |
| 5.2.1.6  | SYNC Time-out  | 5-6  |
| 5.2.1.7  | SYNC Error Indication                                    | 5-6  |
| 5.2.1.8  | LFRAME# Usage  | 5-7  |
| 5.2.1.9  | I/O Cycles   | 5-8  |
| 5.2.1.10 | Bus Master Cycles  | 5-8  |
| 5.2.1.11 | Configuration and ICH Implications                       | 5-8  |
| 5.3      | DMA Operation (D31:F0)                                   | 5-8  |
| 5.3.1    | Channel Priority   | 5-9  |
| 5.3.2    | Address Compatibility Mode                               | 5-10 |
| 5.3.3    | Summary of DMA Transfer Sizes                            | 5-10 |
| 5.3.4    | Autoinitialize   | 5-11 |
| 5.3.5    | Software Commands  | 5-11 |
| 5.4      | PCI DMA  | 5-12 |
| 5.4.1    | PCI DMA Expansion Protocol                               | 5-12 |
| 5.4.2    | PCI DMA Expansion Cycles                                 | 5-13 |
| 5.4.3    | DMA Addresses  | 5-14 |
| 5.4.4    | DMA Data Generation                                      | 5-14 |
| 5.4.5    | DMA Byte Enable Generation                               | 5-14 |
| 5.4.6    | DMA Cycle Termination                                    | 5-15 |
| 5.5      | LPC DMA  | 5-15 |
| 5.5.1    | Asserting DMA Requests                                   | 5-15 |
| 5.5.2    | Abandoning DMA Requests                                  | 5-16 |
| 5.5.3    | General Flow of DMA Transfers                            | 5-16 |
| 5.5.4    | Terminal Count   | 5-17 |
| 5.5.5    | Verify Mode  | 5-17 |
| 5.5.6    | DMA Request De-Assertion                                 | 5-17 |
| 5.5.7    | SYNC Field / LDRQ# Rules                                 | 5-18 |
| 5.6      | 8254 Timers (D31:F0)                                     | 5-19 |
| 5.6.1    | Timer Programming  | 5-19 |
| 5.6.2    | Reading from the Interval Timer                          | 5-20 |
| 5.7      | 8259 Interrupt Controllers (PIC) (D31:F0)                | 5-22 |
| 5.7.1    | Interrupt Handling                                       | 5-22 |
| 5.7.2    | Initialization Command Words (ICWx)                      | 5-24 |
| 5.7.3    | Operation Command Words (OCW)                            | 5-25 |
| 5.7.4    | Modes of Operation                                       | 5-25 |
| 5.7.5    | Masking Interrupts                                       | 5-28 |
| 5.7.6    | Steering PCI Interrupts                                  | 5-28 |
| 5.8      | Advanced Interrupt Controller (APIC) (D31:F0)            | 5-29 |
| 5.8.1    | Interrupt Handling                                       | 5-29 |
| 5.8.2    | Interrupt Mapping  | 5-29 |
| 5.8.3    | APIC Bus Functional Description                          | 5-30 |
| 5.8.3.1  | Physical Characteristics of APIC                         | 5-30 |
| 5.8.3.2  | APIC Bus Arbitration                                     | 5-30 |
| 5.8.3.3  | Bus Message Formats                                      | 5-31 |

|          |   |      |
|----------|---|------|
| 5.9      | Serial Interrupt (D31:F0)   | 5-36 |
| 5.9.1    | Start Frame   | 5-36 |
| 5.9.2    | Data Frames   | 5-36 |
| 5.9.3    | Stop Frame  | 5-37 |
| 5.9.4    | Specific Interrupts not Supported via SERIRQ                                  | 5-37 |
| 5.9.5    | Data Frame Format   | 5-38 |
| 5.10     | Real Time Clock (D31:F0)  | 5-39 |
| 5.10.1   | Update Cycles   | 5-39 |
| 5.10.2   | Interrupts  | 5-40 |
| 5.10.3   | Lockable RAM Ranges   | 5-40 |
| 5.10.4   | Century Rollover  | 5-40 |
| 5.10.5   | Clearing Battery-Backed RTC RAM   | 5-40 |
| 5.11     | Processor Interface (D31:F0)  | 5-42 |
| 5.11.1   | Processor Interface Signals   | 5-42 |
| 5.11.2   | Dual Processor Issues (ICH: 82801AA only)                                     | 5-44 |
| 5.11.2.1 | Signal Differences (ICH: 82801AA only)  | 5-44 |
| 5.11.2.2 | Power Management (ICH: 82801AA only)  | 5-44 |
| 5.11.3   | Speed Strapping for Processor   | 5-45 |
| 5.12     | Power Management (D31:F0)   | 5-46 |
| 5.12.1   | ICH and System Power States   | 5-46 |
| 5.12.2   | System Power Planes   | 5-47 |
| 5.12.3   | SMI#/SCI Generation   | 5-48 |
| 5.12.4   | Dynamic System Clock Control  | 5-49 |
| 5.12.4.1 | Entering/Exiting the C1 State   | 5-49 |
| 5.12.4.2 | Entering/Exiting the C2 State   | 5-50 |
| 5.12.4.3 | Throttling Using STPCLK#  | 5-50 |
| 5.12.4.4 | Transition Rules Among S0/Cx and Throttling States                            | 5-51 |
| 5.12.5   | Sleep States  | 5-51 |
| 5.12.5.1 | Sleep State Overview  | 5-51 |
| 5.12.5.2 | Initiating Sleep State  | 5-51 |
| 5.12.5.3 | Exiting Sleep States  | 5-52 |
| 5.12.5.4 | Sx-G3-Sx, Handling Power Failures   | 5-53 |
| 5.12.6   | Thermal Management  | 5-54 |
| 5.12.6.1 | THRM# Signal  | 5-54 |
| 5.12.6.2 | THRM# Initiated Passive Cooling   | 5-54 |
| 5.12.6.3 | Processor Initiated Passive Cooling<br>(Via Programmed Duty Cycle on STPCLK#) | 5-55 |
| 5.12.6.4 | Active Cooling  | 5-55 |
| 5.12.7   | Event Input Signals and Their Usage   | 5-55 |
| 5.12.7.1 | PWRBTN#—Power Button  | 5-55 |
| 5.12.7.2 | RI#—Ring Indicate   | 5-56 |
| 5.12.7.3 | PME#—PCI Power Management Event   | 5-56 |
| 5.12.8   | Alt Access Mode   | 5-56 |
| 5.12.8.1 | Write Only Registers with Read Paths in Alternate<br>Access Mode              | 5-57 |
| 5.12.8.2 | PIC Reserved Bits   | 5-58 |
| 5.12.8.3 | Read Only Registers with Write Paths in Alternate<br>Access Mode              | 5-59 |
| 5.12.9   | System Power Supplies, Planes, and Signals                                    | 5-59 |
| 5.12.10  | Clock Generators  | 5-60 |
| 5.12.11  | Legacy Power Management Theory of Operation                                   | 5-60 |





|          |   |      |
|----------|---|------|
| 5.13     | System Management (D31:F0)  | 5-61 |
| 5.13.1   | Overview of System Management Functions                             | 5-61 |
| 5.13.2   | Theory of Operation   | 5-61 |
| 5.13.2.1 | Handling an ECC Error or Other Memory Error                         | 5-62 |
| 5.13.3   | Alert on LAN* (ICH: 82801AA only)                                   | 5-62 |
| 5.14     | IDE Controller (D31:F1)   | 5-65 |
| 5.14.1   | PIO Transfers   | 5-66 |
| 5.14.1.1 | PIO IDE Timing Modes  | 5-66 |
| 5.14.1.2 | IORDY Masking   | 5-67 |
| 5.14.1.3 | PIO 32 Bit IDE Data Port Accesses                                   | 5-67 |
| 5.14.1.4 | PIO IDE Data Port Prefetching and Posting                           | 5-67 |
| 5.14.2   | Bus Master Function   | 5-67 |
| 5.14.2.1 | Physical Region Descriptor Format                                   | 5-67 |
| 5.14.2.2 | Line Buffer   | 5-68 |
| 5.14.2.3 | Bus Master IDE Timings  | 5-68 |
| 5.14.2.4 | Interrupts  | 5-69 |
| 5.14.2.5 | Bus Master IDE Operation  | 5-69 |
| 5.14.2.6 | Error Conditions  | 5-70 |
| 5.14.2.7 | 8237 Like Protocol  | 5-70 |
| 5.14.3   | Ultra ATA/33 Protocol   | 5-71 |
| 5.14.3.1 | Signal Descriptions   | 5-71 |
| 5.14.3.2 | Operation   | 5-72 |
| 5.14.3.3 | CRC Calculation   | 5-72 |
| 5.14.3.4 | Synchronous DMA Timings   | 5-72 |
| 5.14.3.5 | Ultra ATA/66 (ICH: 82801AA only)                                    | 5-73 |
| 5.15     | USB Controller (Device 31:Function 2)                               | 5-73 |
| 5.15.1   | Data Structures in Main memory                                      | 5-73 |
| 5.15.1.1 | Frame List Pointer  | 5-73 |
| 5.15.1.2 | Transfer Descriptor (TD)  | 5-74 |
| 5.15.1.3 | Queue Head (QH)   | 5-78 |
| 5.15.2   | Data Transfers To/From Main Memory                                  | 5-79 |
| 5.15.2.1 | Executing the Schedule  | 5-79 |
| 5.15.2.2 | Processing Transfer Descriptors                                     | 5-79 |
| 5.15.2.3 | Command Register, Status Register, and TD<br>Status Bit Interaction | 5-80 |
| 5.15.2.4 | Transfer Queuing  | 5-81 |
| 5.15.3   | Data Encoding and Bit Stuffing                                      | 5-84 |
| 5.15.4   | Bus Protocol  | 5-84 |
| 5.15.4.1 | Bit Ordering  | 5-84 |
| 5.15.4.2 | SYNC Field  | 5-84 |
| 5.15.4.3 | Packet Field Formats  | 5-85 |
| 5.15.4.4 | Address Fields  | 5-86 |
| 5.15.4.5 | Frame Number Field  | 5-86 |
| 5.15.4.6 | Data Field  | 5-86 |
| 5.15.4.7 | Cyclic Redundancy Check (CRC)                                       | 5-86 |
| 5.15.5   | Packet Formats  | 5-87 |
| 5.15.5.1 | Token Packets   | 5-87 |
| 5.15.5.2 | Start of Frame Packets  | 5-87 |
| 5.15.5.3 | Data Packets  | 5-88 |
| 5.15.5.4 | Handshake Packets   | 5-88 |
| 5.15.5.5 | Handshake Responses   | 5-88 |

|        |  |       |
|--------|--|-------|
| 5.15.6 | USB Interrupts .....   | 5-89  |
|        | 5.15.6.1 Transaction Based Interrupts.....                             | 5-89  |
|        | 5.15.6.2 Non-Transaction Based Interrupts.....                         | 5-91  |
| 5.15.7 | USB Power Management .....   | 5-92  |
| 5.15.8 | USB Legacy Keyboard Operation.....                                     | 5-92  |
| 5.16   | SMBus Controller Functional Description (D31:F3).....                  | 5-95  |
| 5.16.1 | Host Controller.....   | 5-95  |
|        | 5.16.1.1 Command Protocols .....                                       | 5-95  |
|        | 5.16.1.2 I <sup>2</sup> C Behavior .....                               | 5-102 |
| 5.16.2 | Bus Arbitration .....  | 5-103 |
| 5.16.3 | Interrupts / SMI# .....  | 5-103 |
| 5.16.4 | SMBALERT# .....  | 5-103 |
| 5.17   | ICH AC'97 2.1 Functional Description .....                             | 5-104 |
| 5.17.1 | AC-link Overview .....   | 5-105 |
| 5.17.2 | AC-link Output Frame (SDOUT) .....                                     | 5-107 |
| 5.17.3 | AC-link Input Frame (SDIN).....  | 5-109 |
| 5.17.4 | Register Access.....   | 5-112 |
| 5.18   | AC-Link Low Power Mode .....   | 5-113 |
| 5.18.1 | External Wake Event .....  | 5-114 |
| 5.18.2 | AC'97 Cold Reset .....   | 5-114 |
| 5.18.3 | AC'97 Warm Reset.....  | 5-114 |
| 5.18.4 | System Reset .....   | 5-115 |
| 6      | Register and Memory Mapping.....                                       | 6-1   |
| 6.1    | PCI Devices and Functions .....  | 6-2   |
| 6.1.1  | PCI Configuration Map .....  | 6-2   |
| 6.1.2  | Standard PCI Bus Configuration Mechanism .....                         | 6-3   |
| 6.2    | I/O Map.....   | 6-4   |
| 6.2.1  | Fixed I/O Address Ranges.....  | 6-4   |
| 6.2.2  | Variable I/O Decode Ranges .....                                       | 6-6   |
| 6.3    | Memory Map.....  | 6-7   |
| 7      | Hub Interface-to-PCI Bridge Registers (D30:F0).....                    | 7-1   |
| 7.1    | PCI Configuration Registers (D30:F0).....                              | 7-1   |
| 7.1.1  | VID—Vendor ID Register (HUB-PCI—D30:F0) .....                          | 7-2   |
| 7.1.2  | DID—Device ID Register (HUB-PCI—D30:F0).....                           | 7-2   |
| 7.1.3  | CMD—Command Register (HUB-PCI—D30:F0).....                             | 7-3   |
| 7.1.4  | PD_STS—Primary Device Status Register (HUB-PCI—D30:F0)....             | 7-4   |
| 7.1.5  | RID—Revision ID Register (HUB-PCI—D30:F0).....                         | 7-5   |
| 7.1.6  | SCC—Sub-Class Code Register (HUB-PCI—D30:F0).....                      | 7-5   |
| 7.1.7  | BCC—Base-Class Code Register (HUB-PCI—D30:F0).....                     | 7-5   |
| 7.1.8  | PMLT—Primary Master Latency Timer Register<br>(HUB-PCI—D30:F0) .....   | 7-5   |
| 7.1.9  | HEADTYP—Header Type Register (HUB-PCI—D30:F0).....                     | 7-6   |
| 7.1.10 | PBUS_NUM—Primary Bus Number Register<br>(HUB-PCI—D30:F0) .....         | 7-6   |
| 7.1.11 | SBUS_NUM—Secondary Bus Number Register<br>(HUB-PCI—D30:F0) .....       | 7-6   |
| 7.1.12 | SUB_BUS_NUM—Subordinate Bus Number Register<br>(HUB-PCI—D30:F0) .....  | 7-6   |
| 7.1.13 | SMLT—Secondary Master Latency Timer Register<br>(HUB-PCI—D30:F0) ..... | 7-7   |



|        |  |      |
|--------|--|------|
| 7.1.14 | IOBASE—I/O Base Register (HUB-PCI—D30:F0) .....                            | 7-7  |
| 7.1.15 | IOLIM—I/O Limit Register (HUB-PCI—D30:F0) .....                            | 7-7  |
| 7.1.16 | SECSTS—Secondary Status Register (HUB-PCI—D30:F0) .....                    | 7-8  |
| 7.1.17 | MEMBASE—Memory Base Register (HUB-PCI—D30:F0) .....                        | 7-9  |
| 7.1.18 | MEMLIM—Memory Limit Register (HUB-PCI—D30:F0) .....                        | 7-9  |
| 7.1.19 | PREF_MEM_BASE—Prefetchable Memory Base Register<br>(HUB-PCI—D30:F0) .....  | 7-9  |
| 7.1.20 | PREF_MEM_MLT—Prefetchable Memory Limit Register<br>(HUB-PCI—D30:F0) .....  | 7-10 |
| 7.1.21 | IOBASE_HI—I/O Base Upper 16 Bits Register<br>(HUB-PCI—D30:F0) .....        | 7-10 |
| 7.1.22 | IOLIM_HI—I/O Limit Upper 16 Bits Register<br>(HUB-PCI—D30:F0) .....        | 7-10 |
| 7.1.23 | INT_LINE—Interrupt Line Register (HUB-PCI—D30:F0) .....                    | 7-10 |
| 7.1.24 | BRIDGE_CNT—Bridge Control Register (HUB-PCI—D30:F0) .....                  | 7-11 |
| 7.1.25 | CNF—ICH Configuration Register (HUB-PCI—D30:F0) .....                      | 7-12 |
| 7.1.26 | MTT—Multi-Transaction Timer Register (HUB-PCI—D30:F0) .....                | 7-13 |
| 7.1.27 | PCI_MAST_STS—PCI Master Status Register<br>(HUB-PCI—D30:F0) .....          | 7-13 |
| 7.1.28 | ERR_CMD—Error Command Register (HUB-PCI—D30:F0) .....                      | 7-14 |
| 7.1.29 | ERR_STS—Error Status Register (HUB-PCI—D30:F0) .....                       | 7-14 |
| 8      | LPC Interface Bridge Registers (D31:F0) .....                              | 8-1  |
| 8.1    | PCI Configuration Registers (D31:F0) .....                                 | 8-1  |
| 8.1.1  | VID—Vendor ID Register (LPC I/F—D31:F0) .....                              | 8-2  |
| 8.1.2  | DID—Device ID Register (LPC I/F—D31:F0) .....                              | 8-2  |
| 8.1.3  | PCICMD—PCI COMMAND Register (LPC I/F—D31:F0) .....                         | 8-3  |
| 8.1.4  | PCISTA—PCI Device Status (LPC I/F—D31:F0) .....                            | 8-4  |
| 8.1.5  | RID—Revision ID Register (LPC I/F—D31:F0) .....                            | 8-5  |
| 8.1.6  | PI—Programming Interface (LPC I/F—D31:F0) .....                            | 8-5  |
| 8.1.7  | SCC—Sub-Class Code Register (LPC I/F—D31:F0) .....                         | 8-5  |
| 8.1.8  | BCC—Base-Class Code Register (LPC I/F—D31:F0) .....                        | 8-5  |
| 8.1.9  | HEADTYP—Header Type Register (LPC I/F—D31:F0) .....                        | 8-5  |
| 8.1.10 | PMBASE—ACPI Base Address (LPC I/F—D31:F0) .....                            | 8-6  |
| 8.1.11 | ACPI_CNTL—ACPI Control (LPC I/F — D31:F0) .....                            | 8-6  |
| 8.1.12 | BIOS_CNTL (LPC I/F—D31:F0) .....   | 8-7  |
| 8.1.13 | TCO_CNTL — TCO Control (LPC I/F — D31:F0) .....                            | 8-7  |
| 8.1.14 | GPIOBASE—GPIO Base Address (LPC I/F—D31:F0) .....                          | 8-8  |
| 8.1.15 | GPIO_CNTL—GPIO Control (LPC I/F—D31:F0) .....                              | 8-8  |
| 8.1.16 | PIRQ[n]_ROUT—PIRQ[A,B,C,D] Routing Control<br>(LPC I/F—D31:F0) .....       | 8-9  |
| 8.1.17 | SERIRQ_CNTL—Serial IRQ Control (LPC I/F—D31:F0) .....                      | 8-9  |
| 8.1.18 | D31_ERR_CFG—Device 31 Error Config Register<br>(LPC I/F—D31:F0) .....      | 8-10 |
| 8.1.19 | D31_ERR_STS—Device 31 Error Status Register<br>(LPC I/F—D31:F0) .....      | 8-10 |
| 8.1.20 | PCI_DMA_CFG—PCI DMA Configuration (LPC I/F—D31:F0) .....                   | 8-11 |
| 8.1.21 | GEN_CNTL — General Control Register (LPC I/F — D31:F0) .....               | 8-11 |
| 8.1.22 | GEN_STA—General Status (LPC I/F—D31:F0) .....                              | 8-13 |
| 8.1.23 | RTC_CONF—RTC Configuration Register (LPC I/F—D31:F0) .....                 | 8-14 |
| 8.1.24 | COM_DEC—LPC I/F Communication Port Decode Ranges<br>(LPC I/F—D31:F0) ..... | 8-14 |

|        |   |      |
|--------|---|------|
| 8.1.25 | FDD/LPT_DEC—LPC I/F FDD & LPT Decode Ranges<br>(LPC I/F—D31:F0).....  | 8-15 |
| 8.1.26 | SND_DEC—LPC I/F Sound Decode Ranges<br>(LPC I/F—D31:F0).....          | 8-15 |
| 8.1.27 | GEN1_DEC—LPC I/F Generic Decode Range 1<br>(LPC I/F—D31:F0).....      | 8-16 |
| 8.1.28 | GEN2_DEC—LPC I/F Generic Decode Range 2<br>(LPC I/F—D31:F0).....      | 8-16 |
| 8.1.29 | LPC_EN—LPC I/F Enables (LPC I/F—D31:F0).....                          | 8-17 |
| 8.1.30 | FWH_DEC_EN—FWH Decode Enable Register<br>(LPC I/F—D31:F0).....        | 8-18 |
| 8.1.31 | FWH_SEL—FWH Select Register (LPC I/F—D31:F0).....                     | 8-19 |
| 8.1.32 | FUNC_DIS—Function Disable Register (LPC I/F—D31:F0).....              | 8-20 |
| 8.2    | DMA I/O Registers.....  | 8-21 |
| 8.2.1  | DMABASE_CA—DMA Base and Current Address Registers.....                | 8-22 |
| 8.2.2  | DMABASE_CC—DMA Base and Current Count Registers.....                  | 8-23 |
| 8.2.3  | DMAMEM_LP—DMA Memory Low Page Registers.....                          | 8-23 |
| 8.2.4  | DMACMD—DMA Command Register.....                                      | 8-24 |
| 8.2.5  | DMASTA—DMA Status Register.....                                       | 8-24 |
| 8.2.6  | DMA_WRMSK—DMA Write Single Mask Register.....                         | 8-25 |
| 8.2.7  | DMACH_MODE—DMA Channel Mode Register.....                             | 8-25 |
| 8.2.8  | DMA Clear Byte Pointer Register.....                                  | 8-26 |
| 8.2.9  | DMA Master Clear Register.....  | 8-26 |
| 8.2.10 | DMA_CLMSK—DMA Clear Mask Register.....                                | 8-26 |
| 8.2.11 | DMA_WRMSK—DMA Write All Mask Register.....                            | 8-27 |
| 8.3    | Timer I/O Registers.....  | 8-28 |
| 8.3.1  | TCW—Timer Control Word Register.....                                  | 8-28 |
|        | 8.3.1.1 RDBK_CMD—Read Back Command.....                               | 8-29 |
|        | 8.3.1.2 LTCH_CMD—Counter Latch Command.....                           | 8-30 |
| 8.3.2  | SBYTE_FMT—Interval Timer Status Byte Format Register.....             | 8-30 |
| 8.3.3  | Counter Access Ports Register.....                                    | 8-31 |
| 8.4    | 8259 Interrupt Controller (PIC) Registers.....                        | 8-32 |
| 8.4.1  | Interrupt Controller I/O MAP.....                                     | 8-32 |
| 8.4.2  | ICW1—Initialization Command Word 1 Register.....                      | 8-33 |
| 8.4.3  | ICW2—Initialization Command Word 2 Register.....                      | 8-34 |
| 8.4.4  | ICW3—Master Controller Initialization Command<br>Word 3 Register..... | 8-34 |
| 8.4.5  | ICW3—Slave Controller Initialization Command<br>Word 3 Register.....  | 8-35 |
| 8.4.6  | ICW4—Initialization Command Word 4 Register.....                      | 8-35 |
| 8.4.7  | OCW1—Operational Control Word 1 (Interrupt Mask) Register....         | 8-35 |
| 8.4.8  | OCW2—Operational Control Word 2 Register.....                         | 8-36 |
| 8.4.9  | OCW3—Operational Control Word 3 Register.....                         | 8-37 |
| 8.4.10 | ELCR1—Master Controller Edge/Level Triggered Register.....            | 8-38 |
| 8.4.11 | ELCR2—Slave Controller Edge/Level Triggered Register.....             | 8-38 |



|         |   |      |
|---------|---|------|
| 8.5     | Advanced Interrupt Controller (APIC) .....                        | 8-39 |
| 8.5.1   | APIC Register Map .....   | 8-39 |
| 8.5.2   | IND—Index Register .....  | 8-39 |
| 8.5.3   | DAT—Data Register .....   | 8-40 |
| 8.5.4   | ID—Identification Register .....                                  | 8-40 |
| 8.5.5   | VER—Version Register .....  | 8-40 |
| 8.5.6   | ARBID—Arbitration ID Register .....                               | 8-41 |
| 8.5.7   | Redirection Table .....   | 8-41 |
| 8.5.8   | IRQPA—IRQ Pin Assertion Register .....                            | 8-43 |
| 8.5.9   | EOIR—EOI Register .....   | 8-43 |
| 8.6     | Real Time Clock Registers .....                                   | 8-44 |
| 8.6.1   | I/O Register Address Map .....                                    | 8-44 |
| 8.6.2   | Indexed Registers .....   | 8-45 |
| 8.6.2.1 | RTC_REGA—Register A .....   | 8-46 |
| 8.6.2.2 | RTC_REGB—Register B (General Configuration) .....                 | 8-47 |
| 8.6.2.3 | RTC_REGC—Register C (Flag Register) .....                         | 8-48 |
| 8.6.2.4 | RTC_REGD—Register D (Flag Register) .....                         | 8-48 |
| 8.7     | Processor Interface Registers .....                               | 8-49 |
| 8.7.1   | NMI_SC—NMI Status and Control Register .....                      | 8-49 |
| 8.7.2   | NMI_EN—NMI Enable (and Real Time Clock Index) .....               | 8-50 |
| 8.7.3   | PORT92—Fast A20 and Init Register .....                           | 8-50 |
| 8.7.4   | COPROC_ERR—Coprocessor Error Register .....                       | 8-50 |
| 8.7.5   | RST_CNT—Reset Control Register .....                              | 8-51 |
| 8.8     | Power Management Registers (D31:F0) .....                         | 8-52 |
| 8.8.1   | Power Management PCI Configuration Registers (D31:F0) .....       | 8-52 |
| 8.8.1.1 | GEN_PMCON_1—General PM Configuration 1 Register (PM—D31:F0) ..... | 8-52 |
| 8.8.1.2 | GEN_PMCON_2—General PM Configuration 2 Register (PM—D31:F0) ..... | 8-53 |
| 8.8.1.3 | GEN_PMCON_3—General PM Configuration 3 Register (PM—D31:F0) ..... | 8-53 |
| 8.8.1.4 | GPI_ROUT—GPI Routing Control Register (PM—D31:F0) .....           | 8-54 |
| 8.8.1.5 | IO_MON_RNG1—IO Monitor Range 1 Register (PM—D31:F0) .....         | 8-54 |
| 8.8.1.6 | IO_MON_RNG2—IO Monitor Range 2 Register (PM—D31:F0) .....         | 8-55 |
| 8.8.1.7 | IO_MON_MSK—IO Monitor Range Mask Register (PM—D31:F0) .....       | 8-55 |
| 8.8.2   | APM I/O Decode .....  | 8-56 |
| 8.8.2.1 | APM_CNT—Advanced Power Management Control Port Register .....     | 8-56 |
| 8.8.2.2 | APM_STS—Advanced Power Management Status Port Register .....      | 8-56 |

|          |   |      |
|----------|---|------|
| 8.8.3    | Power Management I/O Registers .....                                | 8-57 |
| 8.8.3.1  | PM1_STS—Power Management 1 Status Register .....                    | 8-58 |
| 8.8.3.2  | PM1_EN—Power Management 1 Enables Register .....                    | 8-59 |
| 8.8.3.3  | PM1_CNT—Power Management 1 Control .....                            | 8-60 |
| 8.8.3.4  | PM1_TMR—Power Management 1 Timer Register .....                     | 8-60 |
| 8.8.3.5  | PROC_CNT—Processor Control Register .....                           | 8-61 |
| 8.8.3.6  | LV2 — Level 2 Register .....  | 8-62 |
| 8.8.3.7  | GPE0_STS—General Purpose Event 0 Status Register .....              | 8-62 |
| 8.8.3.8  | GPE0_EN—General Purpose Event 0 Enables Register .....              | 8-63 |
| 8.8.3.9  | GPE1_STS—General Purpose Event 1 Status Register .....              | 8-64 |
| 8.8.3.10 | GPE1_EN—General Purpose Event 1 Enable Register .....               | 8-65 |
| 8.8.3.11 | SMI_EN—SMI Control and Enable Register .....                        | 8-65 |
| 8.8.3.12 | SMI_STS—SMI Status Register .....                                   | 8-66 |
| 8.8.3.13 | IOMON_STS_EN — I/O Monitor Status and<br>Enable Register .....      | 8-67 |
| 8.8.3.14 | DEVACT_STS — Device Activity Status Register .....                  | 8-68 |
| 8.8.3.15 | BUS_ADDR_TRACK— Bus Address Tracker .....                           | 8-68 |
| 8.8.3.16 | BUS_CYC_TRACK— Bus Cycle Tracker .....                              | 8-69 |
| 8.9      | System Management TCO Registers (D31:F0) .....                      | 8-70 |
| 8.9.1    | TCO Register I/O Map .....  | 8-70 |
| 8.9.2    | TCO1_RLD—TCO Timer Reload and Current Value .....                   | 8-70 |
| 8.9.3    | TCO1_TMR—TCO Timer Initial Value .....                              | 8-70 |
| 8.9.4    | TCO1_DAT_IN—TCO Data In Register .....                              | 8-71 |
| 8.9.5    | TCO1_DAT_OUT—TCO Data Out Register .....                            | 8-71 |
| 8.9.6    | TCO1_STS—TCO1 Status Register .....                                 | 8-71 |
| 8.9.7    | TCO2_STS—TCO2 Status Register .....                                 | 8-72 |
| 8.9.8    | TCO1_CNT—TCO1 Control Register .....                                | 8-73 |
| 8.9.9    | TCO2_CNT—TCO2 Control Register .....                                | 8-73 |
| 8.9.10   | TCO_MESSAGE1 and TCO_MESSAGE2 Registers<br>(ICH 82801AA only) ..... | 8-74 |
| 8.9.11   | TCO_WDSTATUS—TCO2 Control Register<br>(ICH 82801AA only) .....      | 8-74 |
| 8.10     | General Purpose I/O Registers (D31:F0) .....                        | 8-75 |
| 8.10.1   | GPIO Register I/O Address Map .....                                 | 8-78 |
| 8.10.2   | GPIO_USE_SEL—GPIO Use Select Register .....                         | 8-79 |
| 8.10.3   | GP_IO_SEL—GPIO Input/Output Select Register .....                   | 8-80 |
| 8.10.4   | GP_LVL—GPIO Level for Input or Output Register .....                | 8-81 |
| 8.10.5   | GPO_TTL—GPIO TTL Select Register .....                              | 8-81 |
| 8.10.6   | GPO_BLINK—GPO Blink Enable Register .....                           | 8-82 |
| 8.10.7   | GPI_INV—GPIO Signal Invert Register .....                           | 8-82 |
| 9        | IDE Controller Registers (D31:F1) .....                             | 9-1  |
| 9.1      | PCI Configuration Registers (IDE—D31:F1) .....                      | 9-1  |
| 9.1.1    | VID—Vendor ID Register (IDE—D31:F1) .....                           | 9-2  |
| 9.1.2    | DID—Device ID Register (IDE—D31:F1) .....                           | 9-2  |
| 9.1.3    | CMD—Command Register (IDE—D31:F1) .....                             | 9-2  |
| 9.1.4    | STA—Device Status Register (IDE—D31:F1) .....                       | 9-3  |
| 9.1.5    | RID—Revision ID Register (IDE—D31:F1) .....                         | 9-3  |
| 9.1.6    | PI—Programming Interface (IDE—D31:F1) .....                         | 9-3  |
| 9.1.7    | SCC—Sub Class Code (IDE—D31:F1) .....                               | 9-4  |
| 9.1.8    | BCC—Base Class Code (IDE—D31:F1) .....                              | 9-4  |
| 9.1.9    | MLT—Master Latency Timer (IDE—D31:F1) .....                         | 9-4  |
| 9.1.10   | HEADTYP—Header Type Register (LPC I/F—D31:F0) .....                 | 9-4  |



|         |  |       |
|---------|--|-------|
| 9.1.11  | BM_BASE—Bus Master Base Address Register (IDE—D31:F1) .....              | 9-5   |
| 9.1.12  | IDE_TIM—IDE Timing Register (IDE—D31:F1) .....                           | 9-5   |
| 9.1.13  | SLV_IDETIM—Slave (Drive 1) IDE Timing Register (IDE—D31:F1) .....        | 9-7   |
| 9.1.14  | SDMA_CNT—Synchronous DMA Control Register (IDE—D31:F1) .....             | 9-8   |
| 9.1.15  | SDMA_TIM—Synchronous DMA Timing Register (IDE—D31:F1) .....              | 9-9   |
| 9.1.16  | IDE_CONFIG—IDE I/O Configuration Register .....                          | 9-10  |
| 9.2     | Bus Master IDE I/O Registers (D31:F1) .....                              | 9-12  |
| 9.2.1   | BMIC[P,S]—Bus Master IDE Command Register .....                          | 9-12  |
| 9.2.2   | BMIS[P,S]—Bus Master IDE Status Register .....                           | 9-13  |
| 9.2.3   | BMID[P,S]—Bus Master IDE Descriptor Table Pointer Register .....         | 9-14  |
| 10      | USB Controller Registers (D31:F2) .....                                  | 10-1  |
| 10.1    | PCI Configuration Registers (D31:F2) .....                               | 10-1  |
| 10.1.1  | VID—Vendor Identification Register (USB—D31:F2) .....                    | 10-2  |
| 10.1.2  | DID—Device Identification Register (USB—D31:F2) .....                    | 10-2  |
| 10.1.3  | CMD—Command Register (USB—D31:F2) .....                                  | 10-2  |
| 10.1.4  | STA—Device Status Register (USB—D31:F2) .....                            | 10-3  |
| 10.1.5  | RID—Revision Identification Register (USB—D31:F2) .....                  | 10-3  |
| 10.1.6  | PI—Programming Interface (USB—D31:F2) .....                              | 10-3  |
| 10.1.7  | SCC—Sub Class Code Register (USB—D31:F2) .....                           | 10-4  |
| 10.1.8  | BCC—Base Class Code Register (USB—D31:F2) .....                          | 10-4  |
| 10.1.9  | HEADTYP—Header Type Register (USB—D31:F2) .....                          | 10-4  |
| 10.1.10 | BASE—Base Address Register (USB—D31:F2) .....                            | 10-4  |
| 10.1.11 | INTR_LN—Interrupt Line Register (USB—D31:F2) .....                       | 10-5  |
| 10.1.12 | INTR_PN—Interrupt Pin Register (USB—D31:F2) .....                        | 10-5  |
| 10.1.13 | SB_RELNUM—Serial Bus Release Number Register (USB—D31:F2) .....          | 10-5  |
| 10.1.14 | USB_LEGKEY—USB Legacy Keyboard/Mouse Control Register (USB—D31:F2) ..... | 10-6  |
| 10.1.15 | USB_RES—USB Resume Enable Register (USB—D31:F2) .....                    | 10-7  |
| 10.2    | USB I/O Registers .....  | 10-8  |
| 10.2.1  | USBCMD—USB Command Register .....  | 10-8  |
| 10.2.2  | USBSTA—USB Status Register .....   | 10-11 |
| 10.2.3  | USBINTR—Interrupt Enable Register .....                                  | 10-12 |
| 10.2.4  | FRNUM—Frame Number Register .....  | 10-12 |
| 10.2.5  | FRBASEADD—Frame List Base Address .....                                  | 10-13 |
| 10.2.6  | SOFMOD—Start of Frame Modify Register .....                              | 10-13 |
| 10.2.7  | PORTSC[0,1]—Port Status and Control Register .....                       | 10-14 |

|         |   |       |
|---------|---|-------|
| 11      | SMBus Controller Registers (D31:F3) .....                                       | 11-1  |
| 11.1    | PCI Configuration Registers (SMBUS—D31:F3).....                                 | 11-1  |
| 11.1.1  | VID—Vendor Identification Register (SMBUS—D31:F3).....                          | 11-2  |
| 11.1.2  | DID—Device Identification Register (SMBUS—D31:F3).....                          | 11-2  |
| 11.1.3  | CMD—Command Register (SMBUS—D31:F3).....  | 11-2  |
| 11.1.4  | STA—Device Status Register (SMBUS—D31:F3).....                                  | 11-3  |
| 11.1.5  | RID—Revision ID Register (SMBUS—D31:F3).....                                    | 11-3  |
| 11.1.6  | PI—Programming Interface (SMBUS—D31:F3).....                                    | 11-3  |
| 11.1.7  | SCC—Sub Class Code Register (SMBUS—D31:F3).....                                 | 11-4  |
| 11.1.8  | BCC—Base Class Code Register (SMBUS—D31:F3).....                                | 11-4  |
| 11.1.9  | HEADTYP—Header Type Register (SMBUS—D31:F3).....                                | 11-4  |
| 11.1.10 | SMB_BASE—SMBus Base Address Register<br>(SMBUS—D31:F3) .....                    | 11-4  |
| 11.1.11 | INTR_LN—Interrupt Line Register (SMBUS—D31:F3).....                             | 11-5  |
| 11.1.12 | INTR_PN—Interrupt Pin Register (SMBUS—D31:F3).....                              | 11-5  |
| 11.1.13 | HOSTC—Host Configuration Register (SMBUS—D31:F3).....                           | 11-5  |
| 11.2    | SMBus I/O Registers .....   | 11-6  |
| 11.2.1  | HST_STA—Host Status Register .....  | 11-7  |
| 11.2.2  | HST_CNT—Host Control Register .....   | 11-8  |
| 11.2.3  | HST_CMD—Host Command Register.....  | 11-9  |
| 11.2.4  | XMIT_SLVA—Transmit Slave Address Register.....                                  | 11-9  |
| 11.2.5  | D0—Data 0 Register.....   | 11-9  |
| 11.2.6  | D1—Data 1 Register.....   | 11-9  |
| 11.2.7  | BLOCK_DB—Block Data Byte Register .....   | 11-10 |
| 12      | AC'97 Audio Controller Registers (D31:F5).....                                  | 12-1  |
| 12.1    | AC'97 Audio PCI Configuration Space (D31:F5).....                               | 12-1  |
| 12.1.1  | VID—Vendor Identification Register (Audio—D31:F5).....                          | 12-2  |
| 12.1.2  | DID—Device Identification Register (Audio—D31:F5).....                          | 12-2  |
| 12.1.3  | PCICMD—PCI Command Register (Audio—D31:F5).....                                 | 12-2  |
| 12.1.4  | PCISTA—PCI Device Status Register (Audio—D31:F5).....                           | 12-3  |
| 12.1.5  | RID—Revision Identification Register (Audio—D31:F5).....                        | 12-3  |
| 12.1.6  | PI—Programming Interface Register (Audio—D31:F5).....                           | 12-3  |
| 12.1.7  | SCC—Sub Class Code Register (Audio—D31:F5).....                                 | 12-4  |
| 12.1.8  | BCC—Base Class Code Register (Audio—D31:F5).....                                | 12-4  |
| 12.1.9  | HEADTYP—Header Type Register (Audio—D31:F5).....                                | 12-4  |
| 12.1.10 | NAMBAR—Native Audio Mixer Base Address Register<br>(Audio—D31:F5).....          | 12-5  |
| 12.1.11 | NABMBAR—Native Audio Bus Mastering Base<br>Address Register (Audio—D31:F5)..... | 12-5  |
| 12.1.12 | SVID—Subsystem Vendor ID Register (Audio—D31:F5).....                           | 12-6  |
| 12.1.13 | SID—Subsystem ID Register (Audio—D31:F5).....                                   | 12-6  |
| 12.1.14 | INTR_LN—Interrupt Line Register (Audio—D31:F5).....                             | 12-6  |
| 12.1.15 | INTR_PN—Interrupt Pin Register (Audio—D31:F5).....                              | 12-7  |
| 12.2    | AC'97 Audio I/O Space (D31:F5).....   | 12-7  |
| 12.2.1  | x_BDBAR—Buffer Descriptor Base Address Register .....                           | 12-10 |
| 12.2.2  | x_CIV—Current Index Value Register .....  | 12-10 |
| 12.2.3  | x_LVI—Last Valid Index Register .....   | 12-10 |
| 12.2.4  | x_SR—Status Register .....  | 12-11 |
| 12.2.5  | x_PICB—Position In Current Buffer Register .....                                | 12-12 |
| 12.2.6  | x_PIV—Prefetched Index Value Register .....                                     | 12-12 |





|    |         |   |       |
|----|---------|---|-------|
|    | 12.2.7  | x_CR—Control Register .....                                     | 12-13 |
|    | 12.2.8  | GLOB_CNT—Global Control Register.....                           | 12-14 |
|    | 12.2.9  | GLOB_STA—Global Status Register .....                           | 12-14 |
|    | 12.2.10 | CAS—Codec Access Semaphore Register .....                       | 12-16 |
| 13 |         | AC '97 Modem Controller Registers (D31:F6) .....                | 13-1  |
|    | 13.1    | AC '97 Modem PCI Configuration Space (D31:F6) .....             | 13-1  |
|    | 13.1.1  | VID—Vendor Identification Register (Modem—D31:F6) .....         | 13-2  |
|    | 13.1.2  | DID—Device Identification Register (Modem—D31:F6) .....         | 13-2  |
|    | 13.1.3  | PCICMD—PCI Command Register (Modem—D31:F6) .....                | 13-2  |
|    | 13.1.4  | PCISTA—Device Status Register (Modem—D31:F6) .....              | 13-3  |
|    | 13.1.5  | RID—Revision Identification Register (Modem—D31:F6) .....       | 13-3  |
|    | 13.1.6  | PI—Programming Interface Register (Modem—D31:F6) .....          | 13-3  |
|    | 13.1.7  | SCC—Sub Class Code Register (Modem—D31:F6).....                 | 13-4  |
|    | 13.1.8  | BCC—Base Class Code Register (Modem—D31:F6).....                | 13-4  |
|    | 13.1.9  | HEDTYP—Header Type Register (Modem—D31:F6) .....                | 13-4  |
|    | 13.1.10 | MMBAR—Modem Mixer Base Address Register<br>(Modem—D31:F6) ..... | 13-5  |
|    | 13.1.11 | MBAR—Modem Base Address Register (Modem—D31:F6) .....           | 13-5  |
|    | 13.1.12 | SVID—Subsystem Vendor ID (Modem—D31:F6) .....                   | 13-6  |
|    | 13.1.13 | SID—Subsystem ID (Modem—D31:F6) .....                           | 13-6  |
|    | 13.1.14 | INTR_LN—Interrupt Line Register (Modem—D31:F6) .....            | 13-6  |
|    | 13.1.15 | INT_PIN—Interrupt Pin (Modem—D31:F6) .....                      | 13-7  |
|    | 13.2    | AC'97 Modem I/O Space (D31:F6).....                             | 13-8  |
|    | 13.2.1  | x_BDBAR—Buffer Descriptor List Base Address Register .....      | 13-10 |
|    | 13.2.2  | x_CIV—Current Index Value Register .....                        | 13-10 |
|    | 13.2.3  | x_LVI—Last Valid Index Register .....                           | 13-10 |
|    | 13.2.4  | x_SR—Status Register .....                                      | 13-11 |
|    | 13.2.5  | x_PICB—Position In Current Buffer Register .....                | 13-12 |
|    | 13.2.6  | x_PIV—Prefetch Index Value Register .....                       | 13-12 |
|    | 13.2.7  | x_CR—Control Register .....                                     | 13-13 |
|    | 13.2.8  | GLOB_CNT—Global Control Register.....                           | 13-14 |
|    | 13.2.9  | GLOB_STA—Global Status Register .....                           | 13-14 |
|    | 13.2.10 | CAS—Codec Access Semaphore Register .....                       | 13-16 |
| 14 |         | Electrical Characteristics.....                                 | 14-17 |
|    | 14.1    | Absolute Maximum Ratings .....                                  | 14-17 |
|    | 14.2    | Thermal Characteristics .....                                   | 14-17 |
|    | 14.3    | D.C. Characteristics .....                                      | 14-17 |
|    | 14.4    | A.C. Characteristics .....                                      | 14-22 |
|    | 14.5    | Timing Diagrams.....  | 14-32 |
| 15 |         | Pinout and Package Information.....                             | 15-1  |
|    | 15.1    | Pinout Information.....   | 15-1  |
|    | 15.2    | Package Information.....  | 15-8  |
| 16 |         | Testability.....  | 16-1  |
|    | 16.1    | Test Mode Description .....                                     | 16-1  |
|    | 16.2    | Tri-state Mode.....   | 16-2  |
|    | 16.3    | NAND tree Mode.....   | 16-2  |
| A  |         | Register Index.....   | A-1   |

## Revision History

| Revision | Description     | Date       |
|----------|-----------------|------------|
| -001     | Initial Release | April 1999 |

# Introduction

# 1

The ICH is a highly integrated multifunctional I/O Controller Hub that provides the interface to the PCI Bus and integrates many of the functions needed in today's PC platforms. The ICH communicates with the host controller over a dedicated hub interface. There are two versions of the ICH (82801AA: ICH and 82801AB: ICH0). This provides added flexibility in designing cost-effective system solutions. These devices are pin compatible and are in 241-pin packages.

This datasheet describes the 82801AA (ICH) and 82801AB (ICH0) components. Unless otherwise specified, all non-shaded areas describe the functionality of both components. In the non-shaded areas, the term I/O Controller Hub (ICH) refers to both the 82801AB and 82801AA products. Shading, as is shown here, indicates differences between the two components. In the shaded areas ICH refers to the 82801AA and ICH0 refers to the 82801AB.

## 1.1 About this Manual

This datasheet is intended for Original Equipment Manufacturers and BIOS vendors creating ICH-based products. The datasheet assumes a working knowledge of the vocabulary and principles of USB, IDE, AC'97, SMBus, PCI, ACPI and LPC. Although some details of these features are described within this datasheet, refer to the individual industry specifications listed in Table 1-1 for additional details.

**Table 1-1. Industry Specifications**

| Specification | Location  |
|---------------|---|
| LPC           | <a href="http://developer.intel.com/design/pcisets/lpc/">http://developer.intel.com/design/pcisets/lpc/</a>           |
| AC'97         | <a href="http://developer.intel.com/pc-supply/platform/ac97/">http://developer.intel.com/pc-supply/platform/ac97/</a> |
| SMBus         | <a href="http://www.sbs-forum.org/specs.htm">http://www.sbs-forum.org/specs.htm</a>                                   |
| PCI           | <a href="http://pcsig.com/specs.htm">http://pcsig.com/specs.htm</a>   |
| USB           | <a href="http://www.teleport.com/~usb/docs.htm">http://www.teleport.com/~usb/docs.htm</a>                             |
| ACPI          | <a href="http://www.teleport.com/~acpi/">http://www.teleport.com/~acpi/</a>   |

### Chapter 1. Introduction

Chapter 1 introduces the ICH and provides information on datasheet organization.

### Chapter 2. Signal Description

Chapter 2 provides a detailed description of each ICH signal. Signals are arranged according to interface and details are provided as to the drive characteristics (Input/Output, Open Drain, etc.) of the signals.

### Chapter 3. ICH Power Planes and Pin States

Chapter 3 provides a complete list of signals, their associated power well, their logic level in each suspend state, and their logic level before and after reset.

### Chapter 4. ICH and System Clock Domains

Chapter 4 provides a list of each clock domain associated with the ICH in an ICH-based system.

**Chapter 5. Functional Description**

Chapter 5 provides a detailed description of the functions in the ICH. All functions in this datasheet are abbreviated using the following nomenclature; Device:Function. This manual abbreviates devices with D30 and D31 and functions with F0, F1, F2, F3, F4, F5 and F6. For example, Device 31 Function 5 is abbreviated as D31:F5.

**Chapter 6. Register Memory and I/O Address Maps**

Chapter 6 provides an overview of the registers, fixed I/O ranges, variable I/O ranges, and memory ranges decoded by the ICH.

**Chapter 7. Hub Interface to PCI Bridge Registers**

Chapter 7 provides a detail description of all registers that reside in the Hub Link to PCI bridge. This bridge resides at Device 30, Function 0 (D30:F0).

**Chapter 8. LPC Bridge Registers**

Chapter 8 provides a detail description of all registers that reside in the LPC bridge. This bridge resides at Device 31, Function 0 (D31:F0). This function contains registers for many different units within the ICH including DMA, Timers, Interrupts, Processor Interface, GPIO, Power Management, System Management and RTC.

**Chapter 9. IDE Controller Registers**

Chapter 9 provides a detail description of all registers that reside in the IDE controller. This controller resides at Device 31, Function 1 (D31:F1).

**Chapter 10. USB Controller Registers**

Chapter 10 provides a detail description of all registers that reside in the USB controller. This controller resides at Device 31, Function 2 (D31:F2).

**Chapter 11. SMBus Controller Registers**

Chapter 11 provides a detail description of all registers that reside in the SMBus controller. This controller resides at Device 31, Function 3 (D31:F3).

**Chapter 12. AC'97 Audio Controller Registers**

Chapter 12 provides a detail description of all registers that reside in the audio controller. This controller resides at Device 31, Function 5 (D31:F5). Note that this section of the EDS does not include the native audio mixer registers. Accesses to the mixer registers are forwarded over the AC-link to the codec where the registers reside.

**Chapter 13. AC'97 Modem Controller Registers**

Chapter 13 provides a detail description of all registers that reside in the modem controller. This controller resides at Device 31, Function 6 (D31:F6). Note that this section of the EDS does not include the modem mixer registers. Accesses to the mixer registers are forwarded over the AC-link to the codec where the registers reside.

**Chapter 14. Electrical Characteristics**

Chapter 14 provides all AC and DC characteristics including detailed timing diagrams.

**Chapter 15. Pinout and Package Information**

Chapter 15 provides a table of each signal and its ball assignment in the 241 BGA package. A footprint of the ballout is also provided. This chapter also provides the physical dimensions and characteristics of the 241 BGA package.

**Chapter 16. Testability**

Chapter 17 provides details about the implementation of two test modes provided in the ICH; NAND Tree and High-Z.

**Appendix A. Register Index**

Appendix A provides a list of registers and the location within the datasheet where the register description can be found.

## 1.2 Overview

Both ICH (82801AA) and ICH0 (82801AB) provide extensive I/O support. The ICH provides additional support for 6 PCI bus Request/Grant pairs (instead of 4 for the ICH0), increased IDE capability from Ultra ATA/33 to Ultra ATA/66, and Alert On LAN\*.

82801AA (ICH) / 82801AB (ICH0) functions and capabilities include:

- PCI Rev 2.2 compliant with support for 33 MHz PCI operations
- ICH0 Supports up to 4 Req/Gnt pairs (PCI Slots); ICH supports up to 6 Req/Gnt pairs (PCI Slots)
- Power Management Logic Support
- Enhanced DMA Controller, Interrupt Controller and Timer Functions
- Integrated IDE controller (ICH0 supports Ultra ATA/33; ICH supports Ultra ATA/66)
- USB host interface with support for 2 USB ports
- System Management Bus (SMBus) compatible with most I<sup>2</sup>C devices
- AC'97 2.1 Compliant Link for Audio and Telephony CODECs
- Low Pin Count (LPC) interface
- Firmware Hub (FWH) interface support
- Alert On LAN\* (82801AA ICH only)

The ICH incorporates a variety of PCI functions that are divided into two logical devices (30 and 31). Device 30 is the Hub Interface-To-PCI bridge. Device 31 contains all the other PCI functions as shown in Table 1-2.

**Table 1-2. PCI Devices and Functions**

| Device:Function      | Function Description   |
|----------------------|--|
| Device 30:Function 0 | Hub Interface to PCI Bridge  |
| Device 31:Function 0 | PCI to LPC Bridge<br>(includes: DMA, Timers, compatible interrupt controller, APIC, RTC, processor interface control, power management control, System Management control, and GPIO control) |
| Device 31:Function 1 | IDE Controller   |
| Device 31:Function 2 | USB Controller   |
| Device 31:Function 3 | SMBus Controller   |
| Device 31:Function 4 | Reserved   |
| Device 31:Function 5 | AC'97 Audio Controller   |
| Device 31:Function 6 | AC'97 Modem Controller   |

The following sub-sections provide an overview of I/O Controller Hub capabilities.

## Hub Interface to PCI

The hub interface to PCI Bridge implements the buffering and control logic between PCI and the Hub Interface. The ICH PCI interface provides a 33 MHz, Rev. 2.2 compliant implementation. All PCI signals are 5V tolerant, except PME#. The ICH integrates a PCI arbiter that supports up to four (ICH0: 82801AB) or six (ICH: 82801AA) external PCI bus masters in addition to the internal ICH requests.

## IDE Interface (Bus Master capability and synchronous DMA Mode)

The fast IDE interface supports up to four IDE devices providing an interface for IDE hard disks and CD ROMs. Each IDE device can have independent timings. The IDE interface supports PIO IDE transfers up to 14 Mbytes/sec and Bus Master IDE transfers up to 33 Mbytes/sec for the ICH0 (82801AB) and 66 Mbytes/sec for the ICH (82801AA). It does not consume any ISA DMA resources. The IDE interface integrates 16x32-bit buffers for optimal transfers.

The ICH's IDE system contains two independent IDE signal channels. They can be electrically isolated independently. They can be configured to the standard primary and secondary channels (four devices). There are integrated series resistors on the data and control lines (see Section 5.14, "IDE Controller (D31:F1)" on page 5-65 for details).

## Compatibility Modules (DMA Controller, Timer/Counters, Interrupt Controller)

The DMA controller incorporates the logic of two 82C37 DMA controllers, with seven independently programmable channels. Channels [0:3] are hardwired to 8-bit, count-by-byte transfers, and channels [5:7] are hardwired to 16-bit, count-by-word transfers. Any two of the seven DMA channels can be programmed to support fast Type-F transfers.

The ICH supports two types of DMA (LPC and PC/PCI). DMA via LPC is similar to ISA DMA. LPC DMA and PC/PCI DMA use the ICH's DMA controller. The PC/PCI protocol allows PCI-based peripherals to initiate DMA cycles by encoding requests and grants via two PC/PCI REQ#/GNT# pairs.

LPC DMA is handled through the use of the LDRQ# lines from peripherals and special encodings on LAD[3:0] from the host. Single, Demand, Verify, and Increment modes are supported on the LPC interface. Channel's 0 - 3 are 8 bit channels. Channel's 5 - 7 are 16 bit channels. Channel 4 is reserved as a generic bus master request.

The timer/counter block contains three counters that are equivalent in function to those found in one 82C54 programmable interval timer. These three counters are combined to provide the system timer function, and speaker tone. The 14.31818-MHz oscillator input provides the clock source for these three counters.

The ICH provides an ISA-Compatible interrupt controller that incorporates the functionality of two 82C59 interrupt controllers. The two interrupt controllers are cascaded so that 14 external and two internal interrupts are possible. In addition, the ICH supports a serial interrupt scheme.

All of the registers in these modules can be read and restored. This is required to save and restore system state after power has been removed and restored to the circuit.

### **Advanced Programmable Interrupt Controller (APIC)**

In addition to the standard ISA compatible interrupt controller (PIC) described in the previous section, the ICH incorporates the Advanced Programmable Interrupt Controller (APIC). While the standard interrupt controller is intended for use in a uni-processor system, APIC can be used in either a uni-processor or multi-processor system.

### **Enhanced Universal Serial Bus (USB) Controller**

The USB controller provides enhanced support for the Universal Host Controller Interface (UHCI). This includes support that allows legacy software to use a USB-based keyboard and mouse. The ICH is USB Revision 1.1 compliant.

### **RTC**

The ICH contains a Motorola\* MC146818A-compatible real-time clock with 256 bytes of battery-backed RAM. The real-time clock performs two key functions: keeping track of the time of day and storing system data, even when the system is powered down. The RTC operates on a 32.768-kHz crystal and a separate 3V lithium battery that provides up to 7 years of protection.

The RTC also supports two lockable memory ranges. By setting bits in the configuration space, two 8-byte ranges can be locked to read and write accesses. This prevents unauthorized reading of passwords or other system security information.

The RTC also supports a date alarm, that allows for scheduling a wake up event up to 30 days in advance, rather than just 24 hours in advance.

### **GPIO**

Various general purpose inputs and outputs are provided for custom system design. The number of inputs and outputs varies depending on ICH configuration.

### **Enhanced Power Management**

The ICH's power management functions include enhanced clock control, local and global monitoring support for 14 individual devices, and various low-power (suspend) states, such as Suspend-to-DRAM, and Suspend-to-Disk. A hardware-based thermal management circuit permits software-independent entrance to low-power states. The ICH contains full support for the Advanced Configuration and Power Interface (ACPI) Specification.

### **System Management Bus (SMBus)**

The ICH contains an SMBus Host interface that allows the processor to communicate with SMBus slaves. This interface is compatible with most I<sup>2</sup>C devices.

## Manageability

The ICH integrates several functions designed to manage the system and lower the total cost of ownership (TCO) of the system. These system management functions are designed to report errors, diagnose the system, and recover from system lockups without the aid of an external microcontroller.

- **TCO Timer.** The ICH's integrated programmable TCO Timer is used to detect system locks. The first expiration of the timer generates an SMI# that the system can use to recover from a software lock. The second expiration of the timer causes a system reset to recover from a hardware lock.
- **Processor Present Indicator.** The ICH looks for the processor to fetch the first instruction after reset. If the processor does not fetch the first instruction, the ICH will reboot the system at the safe-mode frequency multiplier.
- **ECC Error Reporting.** When detecting an ECC error, the host controller has the ability to send one of several messages to the ICH. The host controller can instruct the ICH to generate either an SMI#, NMI, SERR#, or TCO interrupt.
- **Function Disable.** The ICH provides the ability to disable the following functions: AC'97 Modem, AC'97 Audio, IDE, USB, or SMBus. Once disabled, these functions no longer decode I/O, memory, or PCI configuration space. Also, no interrupts or power management events are generated from the disable functions.
- **Intruder Detect.** The ICH provides an input signal (INTRUDER#) that can be attached to a switch that is activated by the system case being opened. The ICH can be programmed to generate an SMI# or TCO interrupt due to an active INTRUDER# signal.
- **SMBus.** The ICH integrates an SMBus controller that provides an interface to manage peripherals (e.g., serial presence detection (SPD) or RIMMs and thermal sensors).
- **Alert-On-LAN (82801AA only).** The ICH (82801AA only) supports Alert-On-LAN. In response to a TCO event (intruder detect, thermal event, processor not booting) the ICH sends a message over the SMBus. A LAN controller can decode this SMBus message and send a message over the network to alert the network manager.



## AC'97 2.1 Controller

The digital link in the ICH is AC'97 Revision 2.1 compliant, supporting two codecs with independent PCI functions for audio and modem. Microphone input and left and right audio channels are supported for a high-quality two-speaker audio solution. Wake on ring from suspend is also supported with an appropriate modem codec.

The *Audio Codec '97 (AC'97) Specification* defines a digital link that can be used to attach an audio codec (AC), a modem codec (MC), an audio/modem codec (AMC), or both an AC and an MC. The AC'97 Specification defines the interface between the system logic and the audio or modem codec known as the *AC'97 Digital Link*.

The ability to add cost-effective audio and modem solutions as the platform migrates away from ISA is important. In addition, the AC'97 audio and modem components are software configurable. This reduces configuration errors. The ICH's AC'97 (with the appropriate codecs) not only replaces ISA audio and modem functionality, but also improves overall platform integration by incorporating the AC'97 digital link. Using the ICH's integrated AC'97 digital link reduces cost and eases migration from ISA.

By using an audio codec, the AC'97 digital link allows for cost-effective, high-quality, integrated audio using the ICH. In addition, an AC'97 soft modem can be implemented with the use of a modem codec. Several system options exist when implementing AC'97. The ICH's integrated digital link allows two external codecs to be connected to the ICH. The system designer can provide audio with an audio codec, or a modem with a modem codec. For system requiring both audio and a modem, there are two solutions. The audio codec and the modem codec can be integrated into an AMC or separate audio and modem codecs can be connected to the ICH.

Modem implementation for different countries must be considered as telephone systems may vary. By using a split design, the audio codec can be on-board and the modem codec can be placed on a riser. Intel is developing an AC'97 digital link connector. With a single integrated codec (or AMC), both audio and modem can be routed to a connector near the rear panel where the external ports can be located.



# Signal Description

# 2

This section provides a detailed description of each signal. The signals are arranged in functional groups according to their associated interface.

The “#” symbol at the end of the signal name indicates that the active, or asserted state occurs when the signal is at a low voltage level. When “#” is not present, the signal is asserted when at the high voltage level.

The following notations are used to describe the signal type:

|            |                                    |
|------------|------------------------------------|
| <b>I</b>   | Input Pin                          |
| <b>O</b>   | Output Pin                         |
| <b>OD</b>  | Open Drain Output Pin.             |
| <b>I/O</b> | Bi-directional Input / Output Pin. |

## 2.1 Hub Interface

Table 2-1. Hub Interface Signals

| Name                 | Type | Description  |
|----------------------|------|--|
| HL[10:0]             | I/O  | <b>Hub Interface Signals.</b>  |
| HL[11]<br>(ICH only) | I/O  | <b>Hub Interface Signals (82801AA ICH only).</b>   |
| HL_STB               | I/O  | <b>Hub Interface Strobe:</b> One of two differential strobe signals used to transmit and receive data through the hub interface. |
| HL_STB#              | I/O  | <b>Hub Interface Strobe Complement:</b> Second of the two differential strobe signals.   |
| HLCOMP               | I/O  | <b>Hub Interface Compensation:</b> Used for Hub Interface buffer compensation.   |

## 2.2 Firmware Hub Interface

Table 2-2. Firmware Hub Interface Signals

| Name                   | Type | Description                  |
|------------------------|------|------------------------------|
| FWH[3:0] /<br>LAD[3:0] | I/O  | <b>Firmware Hub Signals.</b> |
| FWH[4] /<br>LFRAME#    | I/O  | <b>Firmware Hub Signals</b>  |

## 2.3 PCI Interface

Table 2-3. PCI Interface Signals (Sheet 1 of 3)

| Name              | Type                        | Description  |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |
|-------------------|-----------------------------|--|-------------------|---------------------|---------|-----------------------|---------|---------------|---------|----------|---------|-----------|---------|-------------|---------|--------------|---------|--------------------|---------|---------------------|---------|----------------------|---------|------------------|---------|-----------------------------|
| <b>AD[31:0]</b>   | I/O                         | <b>PCI Address/Data:</b> AD[31:0] is a multiplexed address and data bus. During the first clock of a transaction, AD[31:0] contain a physical address (32 bits). During subsequent clocks, AD[31:0] contain data. The ICH will drive all 0's on AD[31:0] during the address phase of PCI Interrupt Acknowledge cycles.   |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |
| <b>C/BE#[3:0]</b> | I/O                         | <p><b>Bus Command and Byte Enables:</b> The command and byte enable signals are multiplexed on the same PCI pins. During the address phase of a transaction, C/BE[3:0]# define the bus command. During the data phase C/BE[3:0]# define the Byte Enables.</p> <table border="0"> <tr> <td><b>C/BE[3:0]#</b></td> <td><b>Command Type</b></td> </tr> <tr> <td>0 0 0 0</td> <td>Interrupt Acknowledge</td> </tr> <tr> <td>0 0 0 1</td> <td>Special Cycle</td> </tr> <tr> <td>0 0 1 0</td> <td>I/O Read</td> </tr> <tr> <td>0 0 1 1</td> <td>I/O Write</td> </tr> <tr> <td>0 1 1 0</td> <td>Memory Read</td> </tr> <tr> <td>0 1 1 1</td> <td>Memory Write</td> </tr> <tr> <td>1 0 1 0</td> <td>Configuration Read</td> </tr> <tr> <td>1 0 1 1</td> <td>Configuration Write</td> </tr> <tr> <td>1 1 0 0</td> <td>Memory Read Multiple</td> </tr> <tr> <td>1 1 1 0</td> <td>Memory Read Line</td> </tr> <tr> <td>1 1 1 1</td> <td>Memory Write and Invalidate</td> </tr> </table> <p><b>NOTE:</b> All command encodings not shown are reserved. The ICH does not decode reserved values, and, therefore, will not respond if a PCI master generates a cycle using one of the reserved values.</p> | <b>C/BE[3:0]#</b> | <b>Command Type</b> | 0 0 0 0 | Interrupt Acknowledge | 0 0 0 1 | Special Cycle | 0 0 1 0 | I/O Read | 0 0 1 1 | I/O Write | 0 1 1 0 | Memory Read | 0 1 1 1 | Memory Write | 1 0 1 0 | Configuration Read | 1 0 1 1 | Configuration Write | 1 1 0 0 | Memory Read Multiple | 1 1 1 0 | Memory Read Line | 1 1 1 1 | Memory Write and Invalidate |
| <b>C/BE[3:0]#</b> | <b>Command Type</b>         |  |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |
| 0 0 0 0           | Interrupt Acknowledge       |  |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |
| 0 0 0 1           | Special Cycle               |  |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |
| 0 0 1 0           | I/O Read                    |  |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |
| 0 0 1 1           | I/O Write                   |  |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |
| 0 1 1 0           | Memory Read                 |  |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |
| 0 1 1 1           | Memory Write                |  |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |
| 1 0 1 0           | Configuration Read          |  |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |
| 1 0 1 1           | Configuration Write         |  |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |
| 1 1 0 0           | Memory Read Multiple        |  |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |
| 1 1 1 0           | Memory Read Line            |  |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |
| 1 1 1 1           | Memory Write and Invalidate |  |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |
| <b>DEVSEL#</b>    | I/O                         | <b>Device Select:</b> The ICH asserts DEVSEL# to claim a PCI transaction. As an output, the ICH asserts DEVSEL# when a PCI master peripheral attempts an access to an internal ICH register or main memory. As an input, DEVSEL# indicates the response to an ICH-initiated transaction on the PCI bus. DEVSEL# is tri-stated from the leading edge of PCIRST#. DEVSEL# remains tri-stated by the ICH until driven by a Target device.   |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |
| <b>FRAME#</b>     | I/O                         | <b>Cycle Frame:</b> The current Initiator drives FRAME# to indicate the beginning and duration of a PCI transaction. While the initiator asserts FRAME#, data transfers continue. When the initiator negates FRAME#, the transaction is in the final data phase. FRAME# is an input to the ICH when the ICH is the target, and FRAME# is an output from the ICH when the ICH is the Initiator. FRAME# remains tri-stated by the ICH until driven by an Initiator.  |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |
| <b>IRDY#</b>      | I/O                         | <b>Initiator Ready:</b> IRDY# indicates the ICH's ability, as an Initiator, to complete the current data phase of the transaction. It is used in conjunction with TRDY#. A data phase is completed on any clock both IRDY# and TRDY# are sampled asserted. During a write, IRDY# indicates the ICH has valid data present on AD[31:0]. During a read, it indicates the ICH is prepared to latch data. IRDY# is an input to the ICH when the ICH is the Target and an output from the ICH when the ICH is an Initiator. IRDY# remains tri-stated by the ICH until driven by an Initiator.   |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |
| <b>TRDY#</b>      | I/O                         | <b>Target Ready:</b> TRDY# indicates the ICH's ability as a Target to complete the current data phase of the transaction. TRDY# is used in conjunction with IRDY#. A data phase is completed when both TRDY# and IRDY# are sampled asserted. During a read, TRDY# indicates that the ICH, as a Target, has placed valid data on AD[31:0]. During a write, TRDY# indicates the ICH, as a Target is prepared to latch data. TRDY# is an input to the ICH when the ICH is the Initiator and an output from the ICH when the ICH is a Target. TRDY# is tri-stated from the leading edge of PCIRST#. TRDY# remains tri-stated by the ICH until driven by a target.  |                   |                     |         |                       |         |               |         |          |         |           |         |             |         |              |         |                    |         |                     |         |                      |         |                  |         |                             |

Table 2-3. PCI Interface Signals (Sheet 2 of 3)

| Name   | Type | Description   |
|--|------|---|
| <b>STOP#</b>   | I/O  | <b>Stop:</b> STOP# indicates that the ICH, as a Target, is requesting the Initiator to stop the current transaction. STOP# causes the ICH, as an Initiator, to stop the current transaction. STOP# is an output when the ICH is a Target and an input when the ICH is an Initiator. STOP# is tri-stated from the leading edge of PCIRST#. STOP# remains tri-stated until driven by the ICH.   |
| <b>PAR</b>   | I/O  | <b>Calculated/Checked Parity:</b> PAR uses "even" parity calculated on 36 bits, AD[31:0] plus C/BE[3:0]#. "Even" parity means that the ICH counts the number of "1"s within the 36 bits plus PAR and the sum is always even. The ICH always calculates PAR on 36 bits regardless of the valid byte enables. The ICH generates PAR for address and data phases and only guarantees PAR to be valid one PCI clock after the corresponding address or data phase. The ICH drives and tri-states PAR identically to the AD[31:0] lines except that the ICH delays PAR by exactly one PCI clock. PAR is an output during the address phase (delayed one clock) for all ICH initiated transactions. PAR is an output during the data phase (delayed one clock) when the ICH is the Initiator of a PCI write transaction, and when it is the Target of a read transaction. ICH checks parity when it is the Target of a PCI write transaction. If a parity error is detected, the ICH will set the appropriate internal status bits, and has the option to generate an NMI# or SMI#. |
| <b>PERR# /<br/>GPIO[7]<br/>(ICH only)</b>                              | I/O  | <b>Parity Error (82801AA ICH only):</b> An external PCI device drives PERR# when it receives data that has a parity error. For the 82801AA, the ICH drives PERR# when it detects a parity error. The ICH can either generate an NMI# or SMI# upon detecting a parity error (either detected internally or reported via the PERR# signal).<br>If this signal is not used for PERR#, it can instead be used as a GPIO.<br><br><b>NOTE:</b> GPIO[7] is a non-multiplexed signal on the ICH0 (82801AB).   |
| <b>REQ[0:3]#</b>   | I    | <b>PCI Requests (REQ[0:3]):</b><br><br><b>NOTE:</b> Note:REQ[0]# is programmable to have improved arbitration latency for supporting PCI-based 1394 controllers.  |
| <b>REQ[4]#<br/>REQ[5]# /<br/>REQ[B]# /<br/>GPIO[1]<br/>(ICH only)</b>  | I    | <b>PCI Requests (REQ[4,5]) (82801AA ICH only):</b> REQ[5]# is muxed with PC/PCI REQ[B]# (must choose one or the other, but not both). If not used for PCI or PC/PCI, REQ[5]#/REQ[B]# can instead be used as GPIO[1].<br><br><b>NOTES:</b><br>1. REQ[B]# and GPIO[1] are on both the ICH (82801AA) and ICH0 (82801AB) components.<br>2. GPIO[1] is only multiplexed with REQ[B]# on the ICH0 (82801AB).  |
| <b>GNT[0:3]#</b>   | O    | <b>PCI Grants:</b> Pullup resistors are not required on these signals. If pullups are used, they should be tied to the Vcc3_3 power rail.   |
| <b>GNT[4]#<br/>GNT[5]#/<br/>GNT[B]# /<br/>GPIO[17]#<br/>(ICH only)</b> | O    | <b>PCI Grants (GNT[4,5]) (82801AA ICH only):</b> Pullup resistors are not required on these signals. If pullups are used, they should be tied to the Vcc3_3 power rail.<br>GNT[5]# is muxed with PC/PCI GNT[B]# (must choose one or the other, but not both). If not needed for PCI or PC/PCI, GNT[5]# can instead be used as a GPIO.<br><br><b>NOTES:</b><br>1. GNT[B]# and GPIO[17] are on both the ICH (82801AA) and ICH0 (82801AB) components.<br>2. GPIO[17] is only multiplexed with GNT[B]# on the ICH0 (82801AB).   |
| <b>PCICLK</b>  | I    | <b>PCI Clock:</b> 33 MHz clock. PCICLK provides timing for all transactions on the PCI Bus.   |
| <b>PCIRST#</b>   | O    | <b>PCI Reset:</b> ICH asserts PCIRST# to reset devices that reside on the PCI bus. The ICH asserts PCIRST# during power-up and when S/W initiates a hard reset sequence through the RC (CF9h) register. The ICH drives PCIRST# inactive a minimum of 1 ms after PWROK is driven active. The ICH drives PCIRST# active a minimum of 1 ms when initiated through the RC register.   |
| <b>PLOCK#</b>  | I/O  | <b>PCI Lock:</b> Indicates an exclusive bus operation and may require multiple transactions to complete. ICH asserts PLOCK# when it performs non-exclusive transactions on the PCI bus. PLOCK# is ignored when PCI masters are granted the bus.   |

Table 2-3. PCI Interface Signals (Sheet 3 of 3)

| Name  | Type | Description   |
|---|------|---|
| <b>SERR#</b>  | I    | <b>System Error:</b> SERR# can be pulsed active by any PCI device that detects a system error condition. Upon sampling SERR# active, the ICH has the ability to generate an NMI, SMI#, or interrupt.  |
| <b>PME#</b>   | I    | <b>PCI Power Management Event:</b> PCI peripherals drive PME# to wake the system from low-power states S1-S5.   |
| <b>REQ[A]# /</b><br>GPIO[0]<br><b>REQ[B]# /</b><br>REQ[5]# /<br>GPIO[1]   | I    | <b>PC/PCI DMA Request [A:B]:</b> This request serializes ISA-like DMA Requests for the purpose of running ISA-compatible DMA cycles over the PCI bus. This is used by devices such as PCI based Super I/O or audio codecs which need to perform legacy 8237 DMA but have no ISA bus.<br><br>When not used for PC/PCI requests, these signals can be used as General Purpose Inputs.<br><b>NOTES:</b><br>1. For the 82810AA, REQ[B]# can instead be used as the 6th PCI bus request.<br>2. REQ[5]# (shown in column 1) is only available on the ICH (82801AA). |
| <b>GNT[A]# /</b><br>GPIO[16]<br><b>GNT[B]# /</b><br>GNT[5]# /<br>GPIO[17] | O    | <b>PC/PCI DMA Acknowledges [A: B]:</b> This grant serializes an ISA-like DACK# for the purpose of running DMA/ISA Master cycles over the PCI bus. This is used by devices such as PCI based Super/I/O or audio codecs that need to perform legacy 8237 DMA but have no ISA bus.<br><br>When not used for PC/PCI, these signals can be used as General Purpose Outputs. .<br><b>NOTES:</b><br>1. For the 82810AA, GNTB# can also be used as the 6th PCI bus master grant output.<br>2. GNT[5]# (shown in column 1) is only available on the ICH (82801AA).     |

## 2.4 IDE Interface

Table 2-4. IDE Interface Signals (Sheet 1 of 2)

| Name                                  | Type | Description  |
|---------------------------------------|------|--|
| <b>PDCS1#,</b><br><b>SDCS1#</b>       | O    | <b>Primary and Secondary IDE Device Chip Selects for 100 Range:</b> For ATA command register block. This output signal is connected to the corresponding signal on the primary or secondary IDE connector.   |
| <b>PDCS3#,</b><br><b>SDCS3#</b>       | O    | <b>Primary and Secondary IDE Device Chip Select for 300 Range:</b> For ATA control register block. This output signal is connected to the corresponding signal on the primary or secondary IDE connector.  |
| <b>PDA[2:0],</b><br><b>SDA[2:0]</b>   | O    | <b>Primary and Secondary IDE Device Address:</b> These output signals are connected to the corresponding signals on the primary or secondary IDE connectors. They are used to indicate which byte in either the ATA command block or control block is being addressed.   |
| <b>PDD[15:0],</b><br><b>SDD[15:0]</b> | I/O  | <b>Primary and Secondary IDE Device Data:</b> These signals directly drive the corresponding signals on the primary or secondary IDE connector.  |
| <b>PDDREQ,</b><br><b>SDDREQ</b>       | I    | <b>Primary and Secondary IDE Device DMA Request:</b> These input signals are directly driven from the DRQ signals on the primary or secondary IDE connector. It is asserted by the IDE device to request a data transfer, and used in conjunction with the PCI bus master IDE function and are not associated with any AT compatible DMA channel.  |
| <b>PDDACK#,</b><br><b>SDDACK#</b>     | O    | <b>Primary and Secondary IDE Device DMA Acknowledge:</b> These signals directly drive the DAK# signals on the primary and secondary IDE connectors. Each is asserted by the ICH to indicate to IDE DMA slave devices that a given data transfer cycle (assertion of DIOR# or DIOW#) is a DMA data transfer cycle. This signal is used in conjunction with the PCI bus master IDE function and are not associated with any AT-compatible DMA channel. |

Table 2-4. IDE Interface Signals (Sheet 2 of 2)

| Name   | Type | Description  |
|--|------|--|
| <b>PDIOR# /</b><br>(PDWSTB /<br>PRDMARDY#)<br><br><b>SDIOR# /</b><br>(SDWSTB /<br>SRDMARDY#)   | O    | <p><b>Primary and Secondary Disk I/O Read (PIO and Non-Ultra ATA):</b> This is the command to the IDE device that it may drive data onto the PDD or SDD lines. Data is latched by the ICH on the deassertion edge of PDIOR# or SDIOR#. The IDE device is selected either by the ATA register file chip selects (PDCS1# or SDCS1#, PDCS3# or SDCS3#) and the PDA or SDA lines, or the IDE DMA acknowledge (PDDAK# or SDDAK#).</p> <p><b>Primary and Secondary Disk Write Strobe (Ultra ATA Writes to Disk):</b> This is the data write strobe for writes to disk. When writing to disk, ICH drives valid data on rising and falling edges of PDWSTB or SDWSTB.</p> <p><b>Primary and Secondary Disk DMA Ready (Ultra ATA Reads from Disk):</b> This is the DMA ready for reads from disk. When reading from disk, ICH deasserts PRDMARDY# or SRDMARDY# to pause burst data transfers.</p> |
| <b>PDIOW# /</b><br>(PDSTOP)<br><b>SDIOW# /</b><br>(SDSTOP)                                     | O    | <p><b>Primary and Secondary Disk I/O Write (PIO and Non-Ultra ATA):</b> This is the command to the IDE device that it may latch data from the PDD or SDD lines. Data is latched by the IDE device on the deassertion edge of PDIOW# or SDIOW#. The IDE device is selected either by the ATA register file chip selects (PDCS1# or SDCS1#, PDCS3# or SDCS3#) and the PDA or SDA lines, or the IDE DMA acknowledge (PDDAK# or SDDAK#).</p> <p><b>Primary and Secondary Disk Stop (Ultra ATA):</b> ICH asserts this signal to terminate a burst.</p>  |
| <b>PIORDY /</b><br>(PDRSTB /<br>PWDARMARDY#)<br><br><b>SIORDY /</b><br>(SDRSTB /<br>SWDMARDY#) | I    | <p><b>Primary and Secondary I/O Channel Ready (PIO):</b> This signal will keep the strobe active (PDIOR# or SDIOR# on reads, PDIOW# or SDIOW# on writes) longer than the minimum width. It adds wait states to PIO transfers.</p> <p><b>Primary and Secondary Disk Read Strobe (Ultra ATA Reads from Disk):</b> When reading from disk, ICH latches data on rising and falling edges of this signal from the disk.</p> <p><b>Primary and Secondary Disk DMA Ready (Ultra ATA Writes to Disk):</b> When writing to disk, this is de-asserted by the disk to pause burst data transfers.</p>   |

## 2.5 Low Pin Count (LPC) Interface

Table 2-5. LPC Interface Signals

| Name                                 | Type | Description  |
|--------------------------------------|------|--|
| <b>LAD[3:0] /</b><br><b>FWH[3:0]</b> | I/O  | <b>LPC Multiplexed Command, Address, Data:</b> Internal pull-ups are provided.   |
| <b>LFRAME# /</b><br><b>FWH[4]</b>    | O    | <b>LPC Frame:</b> Indicates the start of an LPC cycle, or an abort.  |
| <b>LDRQ[0]#</b>                      | I    | <b>LPC Serial DMA/Master Request Inputs:</b> Used to request DMA or bus master access. Typically connected to external Super I/O device.             |
| <b>LDRQ[1]# /</b><br><b>GPI0[8]</b>  | I    | <b>LPC Serial DMA/Master Request Inputs:</b> Second DMA or bus master request. If LDRQ[1]# is not needed, it can be used as a General Purpose Input. |

## 2.6 Interrupt Interface

Table 2-6. Interrupt Signals

| Name       | Type | Description   |
|------------|------|---|
| SERIRQ     | I/O  | <b>Serial Interrupt Request:</b> This pin implements the serial interrupt protocol.   |
| PIRQ[A:D]# | I/OD | <b>PCI Interrupt Requests:</b> The PIRQx# signals can be routed to interrupts 3:7, 9:12, 14 or 15 as described in the Interrupt Steering section. Each PIRQx# line has a separate Route Control Register.<br>In APIC mode, these signals are connected to the internal I/O APIC in the following fashion: PIRQA# is connected to IRQ16, PIRQB# to IRQ17, PIRQC# to IRQ18, and PIRQD# to IRQ19. This frees the ISA interrupts. |
| IRQ[14,15] | I    | <b>Interrupt Request 14, 15:</b> These interrupt inputs are connected to the IDE drives. IRQ14 is used by the drives connected to the Primary controller and IRQ15 is used by the drives connected to the Secondary controller.   |
| APICCLK    | I    | <b>APIC Clock:</b> This clock can run at 16.66667 MHz and is the APIC bus clock.  |
| APICD[1:0] | I/OD | <b>APIC Data:</b> These bi-directional open drain signals are used to send and receive data over the APIC bus. As inputs the data is valid on the rising edge of APICCLK. As outputs, new data is driven from the rising edge of the APICCLK.   |

## 2.7 USB Interface

Table 2-7. USB Interface Signals

| Name             | Type | Description   |
|------------------|------|---|
| USBP1P<br>USBP1N | I/O  | <b>Universal Serial Bus Port 1 Differential:</b> Bus Data/Address/Command Bus   |
| USBP0P<br>USBP0N | I/O  | <b>Universal Serial Bus Port 0 Differential:</b> Bus Data/Address/Command Bus   |
| OC[1:0]#         | I    | <b>Overcurrent Indicators:</b> These signals set corresponding bits in the USB controller to indicate that an overcurrent condition has occurred. |



## 2.8 Power Management Interface

Table 2-8. Power Management Interface Signals

| Name   | Type | Description   |
|--|------|---|
| <b>THRM#</b>                                       | I    | <b>Thermal Alarm:</b> Active low signal generated by external hardware to start the Hardware clock throttling mode. Can also generate an SMI# or an SCI.  |
| <b>SLP_S3#/<br/>GPIO[24]</b>                       | O    | <b>S3 Sleep Control:</b> Power plane control. The signal is used to shut power off to all non-critical systems when in S3 (Suspend To RAM) state. It can be used as General Purpose Output if STR is not supported.   |
| <b>SLP_S5#</b>                                     | O    | <b>S5 Sleep Control:</b> Power plane control. The signal is used to shut power off to all non-critical systems when in the S4 (Suspend To Disk) or S5 (Soft Off) states.  |
| <b>PWROK</b>                                       | I    | <b>Power OK:</b> When asserted, PWROK is an indication to the ICH that core power and PCICLK have been stable for at least 1 ms. PWROK can be driven asynchronously. When PWROK is negated, the ICH asserts PCIRST#.  |
| <b>PWRBTN#</b>                                     | I    | <b>Power Button:</b> The Power Button will cause SMI# or SCI to indicate a system request to go to a sleep state. If the system is already in a sleep state, this signal will cause a wake event. If PWRBTN# is pressed for more than 4 seconds, this will cause an unconditional transition (power button override) to the S5 state with only the PWRBTN# available as a wake event. Override will occur, even if the system is in the S1-S4 states.   |
| <b>RI#</b>   | I    | <b>Ring Indicate:</b> From the modem interface. Can be enabled as a wake event, and this is preserved across power failures.  |
| <b>RSMRST#</b>                                     | I    | <b>Resume Well Reset:</b> Used for resetting the resume power plane logic.  |
| <b>SUS_STAT#/<br/>GPIO[25]</b>                     | O    | <b>Suspend Status:</b> This signal is asserted by the ICH to indicate that the system will be entering a low power state soon. This can be monitored by devices with memory that need to switch from normal refresh to suspend refresh mode. It can also be used by other peripherals as an indication that they should isolate their outputs that may be going to powered-off planes. This signal is called LPCPD# on the LPC Interface.<br><br><b>NOTE:</b> Can be used as General Purpose Output if external logic does not require this signal. |
| <b>SUSCLK/<br/>GPIO[26]<br/>(82801AA<br/>only)</b> | O    | <b>Suspend Clock (82801AA only):</b> Output of the RTC generator circuit to use by other chips for refresh clock.<br><br><b>NOTES:</b><br>1. Can be used as General Purpose Output if external logic doesn't require this signal.<br>2. GPIO[26] is a non-multiplexed signal on the ICH0 (82801AB).   |

## 2.9 Processor Interface

Table 2-9. Processor Interface Signals

| Name           | Type | Description   |
|----------------|------|---|
| <b>A20M#</b>   | OD   | <b>Mask A20:</b> A20M# will go active based on either setting the appropriate bit in the Port 92h register, or based on the A20GATE input being active.<br><b>Speed Strap:</b> During the reset sequence, ICH drives A20M# high if the corresponding bit is set in the FREQ_STRP register.  |
| <b>CPUSLP#</b> | OD   | <b>Processor Sleep:</b> This signal puts the processor into a state that saves substantial power compared to Stop-Grant state. However, during that time, no snoops occur. The ICH can optionally assert the processorSLP# signal when going to the S1 state.   |
| <b>FERR#</b>   | I    | <b>Numeric Coprocessor Error:</b> This signal is tied to the coprocessor error signal on the processor. IGNNE# is only used if the ICH coprocessor error reporting function is enabled in the General Control Register (Device 31:Function 0, Offset D0, bit 13). If FERR# is asserted, the ICH generates an internal IRQ13 to its interrupt controller unit. It is also used to gate the IGNNE# signal to ensure that IGNNE# is not asserted to the processor unless FERR# is active. FERR# requires an external weak pull-up to ensure a high level when the coprocessor error function is disabled.  |
| <b>IGNNE#</b>  | OD   | <b>Ignore Numeric Error:</b> This signal is connected to the ignore error pin on the processor. IGNNE# is only used if the ICH coprocessor error reporting function is enabled in the General Control Register (Device 31:Function 0, Offset D0, bit 13). If FERR# is active, indicating a coprocessor error, a write to the Coprocessor Error Register (F0h) causes the IGNNE# to be asserted. IGNNE# remains asserted until FERR# is negated. If FERR# is not asserted when the Coprocessor Error Register is written, the IGNNE# signal is not asserted.<br><b>Speed Strap:</b> During the reset sequence, ICH drives IGNNE# high if the corresponding bit is set in the FREQ_STRP register. |
| <b>INIT#</b>   | OD   | <b>Initialization:</b> INIT# is asserted by the ICH for 16 PCI clocks to reset the processor. The 82810AA can be configured to support Processor BIST. In that case, INIT# will be active when PCIRST# is active.   |
| <b>INTR</b>    | OD   | <b>Processor Interrupt:</b> INTR is released by the ICH to signal the processor that an interrupt request is pending and needs to be serviced. It is an asynchronous output and normally driven low.<br><b>Speed Strap:</b> During the reset sequence, ICH drives INTR high if the corresponding bit is set in the FREQ_STRP register.  |
| <b>NMI</b>     | OD   | <b>Non-Maskable Interrupt:</b> NMI is used to force a non-maskable interrupt to the processor. The ICH can generate an NMI when either SERR# or IOCHK# is asserted. The processor detects an NMI when it detects a rising edge on NMI. NMI is reset by setting the corresponding NMI source enable/disable bit in the NMI Status and Control Register.<br><b>Speed Strap:</b> During the reset sequence, ICH drives NMI high if the corresponding bit is set in the FREQ_STRP register.   |
| <b>SMI#</b>    | OD   | <b>System Management Interrupt:</b> SMI# is an active low output synchronous to PCICLK. It is asserted by the ICH in response to one of many enabled hardware or software events.   |
| <b>STPCLK#</b> | OD   | <b>Stop Clock Request:</b> STPCLK# is an active low output synchronous to PCICLK. It is asserted by the ICH in response to one of many hardware or software events. When the processor samples STPCLK# asserted, it responds by stopping its internal clock. STPCLK# is floated during PCIRST#.   |
| <b>RCIN#</b>   | I    | <b>Keyboard Controller Reset Processor:</b> The keyboard controller can generate INIT# to the processor. This saves the external OR gate with the ICH's other sources of INIT#. When the ICH detects the assertion of this signal, INIT# is generated for 16 PCI clocks.  |
| <b>A20GATE</b> | I    | <b>A20 Gate:</b> From the keyboard controller. Acts as an alternative method to force the A20M# signal active. Saves the external OR gate needed with various other PCIssets.   |

## 2.10 SMBus Interface

This is accessed from the PCI Device 31:Function 3.

**Note:** Although this is called the SMBus logic, it has capability to access slaves that are I<sup>2</sup>C compatible.

Table 2-10. SM Bus Interface Signals

| Name                      | Type | Description   |
|---------------------------|------|---|
| <b>SMBDATA</b>            | I/OD | <b>SMBus Data:</b> External pull-up is required.  |
| <b>SMBCLK</b>             | I/OD | <b>SMBus Clock:</b> External pull-up is required.   |
| <b>SMBALERT#/GPIO[11]</b> | I    | <b>SMBus Alert:</b> This signal is used to wake the system or generate SMI#. If not used for SMBALERT#, it can be used as GPIO[11]. |

## 2.11 System Management Interface

Table 2-11. System Mangement Signal

| Name   | Type | Description   |
|--|------|---|
| <b>INTRUDER#</b>                               | I    | <b>Intruder Detect:</b> Disables system if box detected open. If this signal is not connected to a switch, the software can directly read the state of the input and use it as a GPI.   |
| <b>ALERTCLK</b><br>GPIO[27]<br>(82801AA only)  | I/OD | <b>Alert On LAN* Clock (82801AA only):</b> External pull-up is required<br><br><b>NOTES:</b><br>1. This signal requires a pull-up resistor, even if it is not used.<br>2. GPIO[27] is a non-multiplexed signal on the ICH0 (82801AB). |
| <b>ALERTDATA</b><br>GPIO[28]<br>(82801AA only) | I/OD | <b>Alert On LAN* Data (82801AA only):</b> External pull-up is required<br><br><b>NOTES:</b><br>1. This signal requires a pull-up resistor, even if it is not used.<br>2. GPIO[28] is a non-multiplexed signal on the ICH0 (82801AB).  |

## 2.12 Real Time Clock Interface

Table 2-12. Real Time Clock Interface

| Name         | Type    | Description   |
|--------------|---------|---|
| <b>RTCX1</b> | Special | <b>Crystal Input 1:</b> Connected to the 32.768 KHz crystal. If no external crystal is used, then RTCX1 can be driven with the desired clock rate.  |
| <b>RTCX2</b> | Special | <b>Crystal Input 2:</b> Connected to the 32.768 KHz crystal. If no external crystal is used, then RTCX2 should be left floating.  |
| <b>VBIAS</b> | I       | <b>RTC Bias Voltage:</b> This pin is used to provide a reference voltage. The DC voltage applied to this pin sets a current which is mirrored throughout the oscillator and buffer circuitry. |

## 2.13 Other Clocks

Table 2-13. Other Clocks

| Name  | Type | Description   |
|-------|------|---|
| CLK14 | I    | <b>Oscillator Clock:</b> Used for 8254 timers. Runs at 14.31818 MHz. This clock is permitted to stop during S3 (or lower) states. |
| CLK48 | I    | <b>48 MHz Clock:</b> Used to run the USB controller. Runs at 48 MHz. This clock is permitted to stop during S3 (or lower) states. |
| CLK66 | I    | <b>66 MHz Clock:</b> Used to run the hub interface. Runs at 66 MHz. This clock is permitted to stop during S3 (or lower) states.  |

## 2.14 Miscellaneous Signals

Table 2-14. Miscellaneous Signals

| Name   | Type | Description   |
|--------|------|---|
| SPKR   | O    | <b>Speaker:</b> The SPKR signal is the output of counter 2 and is internally "ANDed" with Port 61h bit 1 to provide Speaker Data Enable. This signal drives an external speaker driver device, which in turn drives the system speaker. Upon PCIRST#, its output state is 1.<br><br>Note: SPKR is sampled at the rising edge of PWROK as a functional strap. See Section 2.18.1 for more details. |
| RTCST# | I    | <b>RTC Reset:</b> When asserted, this signal resets register bits in the RTC well and sets the RTC_PWR_STS bit (bit 2 in GEN_PMC0N3 register). This signal is also used to enter the test modes documented in Section 2.13.2.   |

## 2.15 AC'97 Link

These signals are for Device 31:Functions 5 and 6.

Table 2-15. AC'97 Link Signals

| Name                  | Type | Description  |
|-----------------------|------|--|
| AC_RST#               | O    | <b>AC97 Reset:</b> Master H/W reset to external Codec(s)   |
| AC_SYNC               | O    | <b>AC97 Sync:</b> 48 KHz fixed rate sample sync to the Codec(s)  |
| AC_BIT_CLK            | I    | <b>AC97 Bit Clock:</b> 12.288 MHz serial data clock generated by the external Codec(s)   |
| AC_SDOUT              | O    | <b>AC97 Serial Data Out:</b> Serial TDM data output to the Codec(s)<br>Note: AC_SDOUT is sampled at the rising edge of PWROK as a functional strap. See Section 2.18.1 for more details. |
| AC_SDIN 0             | I    | <b>AC97 Serial Data In 0:</b> Serial TDM data input from a Codec   |
| AC_SDIN 1/<br>GPIO[9] | I    | <b>AC97 Serial Data In 1:</b> Serial TDM data input from a Codec.<br>Note: If a separate codec is not used, this signal can be used as a GPI.  |

## 2.16 General Purpose I/O

Table 2-16. General Purpose I/O Signals

| Name        | Type | Description  |
|-------------|------|--|
| GPIO[0:1]   | I    | Fixed as Input only. Main Power Well. Can instead be used for PC/PCI REQ[A:B]#. For the ICH (82801AA), GPIO[1] can alternatively be used for PCI REQ[5]#.  |
| GPIO[2:4]   | I    | Not implemented  |
| GPIO[5:6]   | I    | Fixed as Input only. Main Power Well.  |
| GPIO[7]     | I    | Fixed as Input only. Main Power Well. For the ICH (82801AA), this signal can instead be used as PERR#.   |
| GPIO[8]     | I    | Fixed as Input only. Resume Power Well. Can instead be used for LDRQ[1]#.  |
| GPIO[9]     | I    | Fixed as Input only. Resume Power Well. Can instead be used for AC_SDIN[1].  |
| GPIO[10]    | I    | Not implemented.   |
| GPIO[11]    | I    | Fixed as Input only. Resume Power Well. Can instead be used for SMBALERT#.   |
| GPIO[12:13] | I    | Fixed as Input only. Resume Power Well.  |
| GPIO[14:15] | I    | Not implemented.   |
| GPIO[16:17] | O    | Fixed as Output only. Main Power Well. Can instead be used for PC/PCI GNT[A:B]#. For the ICH (82801AA), GPIO[17] can alternatively be used for PCI GNT[5]# |
| GPIO[18:20] | O    | Not implemented  |
| GPIO[21:22] | O    | Fixed as Output only. Main Power Well  |
| GPIO[23]    | OD   | Fixed as Output only. Main Power Well.   |
| GPIO[24]    | O    | Fixed as Output only. Resume Power Well. Can instead be used for SLP_S3#   |
| GPIO[25]    | O    | Fixed as Output only. Resume Power Well. Can instead be used for SUS_STAT#.  |
| GPIO[26]    | O    | Fixed as Output only. Resume Power Well. For the ICH (82801AA), this signal can instead be used for SUS_CLK.   |
| GPIO[27:28] | OD   | Can be Input or Output. Resume Power Well. For the ICH (82801AA), this signal can instead be used for ALERTCLK and ALERTDATA.                              |
| GPIO[29:31] | I/O  | Not implemented  |

## 2.17 Power and Ground

Table 2-17. Power and Ground Signals

| Name          | Description   |
|---------------|---|
| Vcc3_3 [16:0] | <b>Power for Core:</b> 3.3V. This power will be shut off in S3, S5, or G3 states.   |
| Vcc1_8 [6:0]  | <b>Power for the Hub Interface:</b> 1.8V. This power will be shut off in S3, S4, or G3 states.  |
| VccSUS        | <b>Power for Resume Well:</b> 3.3V. This power is not expected to be shut off unless the system is unplugged.   |
| VccRTC        | <b>Power for RTC Well:</b> 3.3V. (Although can drop to 2.0V if the system is in G3 state). This power is not expected to be shut off unless the RTC battery is removed or drained.<br>Note: Implementations should not attempt to clear CMOS by using a jumper to pull VccRTC low. Clearing CMOS in an ICH-based platform can be done by using a jumper on RTCRST# or GPI, or using SAFEMODE strap. |
| VBIAS         | <b>RTC Bias Voltage:</b> See Section 2.12, "Real Time Clock Interface" on page 2-9.   |
| 5VREF         | <b>Reference for 5V tolerance on inputs:</b> This power will be shut off in S3, S5, or G3 states.   |
| HUBREF        | <b>Reference for the Hub Interface:</b> 0.9V. This power will be shut off in S3, S4, or G3 states.  |
| Vss[12:0]     | Grounds.  |

## 2.18 Pin Straps

### 2.18.1 Functional Strap

Table 2-18 shows signals that are used for static configuration. They are sampled at the rising edge of PWROK to select configurations, and then revert later to their normal usage. To invoke the associated mode, the signal should be driven at least 4 PCI clocks prior to the time it is sampled.

Table 2-18. Functional Strap Definitions

| Signal   | Usage     | When Sampled         | Comment   |
|----------|-----------|----------------------|---|
| SPKR     | NO REBOOT | Rising Edge of PWROK | The signal has a weak internal pull-up. If the signal is sampled low, this indicates that the system is strapped to the "No Reboot" mode (ICH will disable the TCO Timer system reboot feature). The status of this strap is readable via the NO_REBOOT bit (bit 1, D31: F0, Offset D4h).         |
| AC_SDOUT | SAFE MODE | Rising Edge of PWROK | The signal has a weak internal pull-down. If the signal is sampled high, the ICH will set the processor speed strap pins for safe mode. Refer to processor specification for speed strapping definition. The status of this strap is readable via the SAFE_MODE bit (bit 2, D31: F0, Offset D4h). |

## 2.18.2 Test Straps

When PWROK is active (high), driving RTCRST# low for a number of PCI clocks (33 MHz) will activate a particular test mode as specified in Table 2-19.

**Note:** RTCRST# can be driven anytime after PCIRST# is inactive but it is better to drive it as soon as PWROK goes high to enter a test mode.

Refer to Section 16 for a description of each test mode.

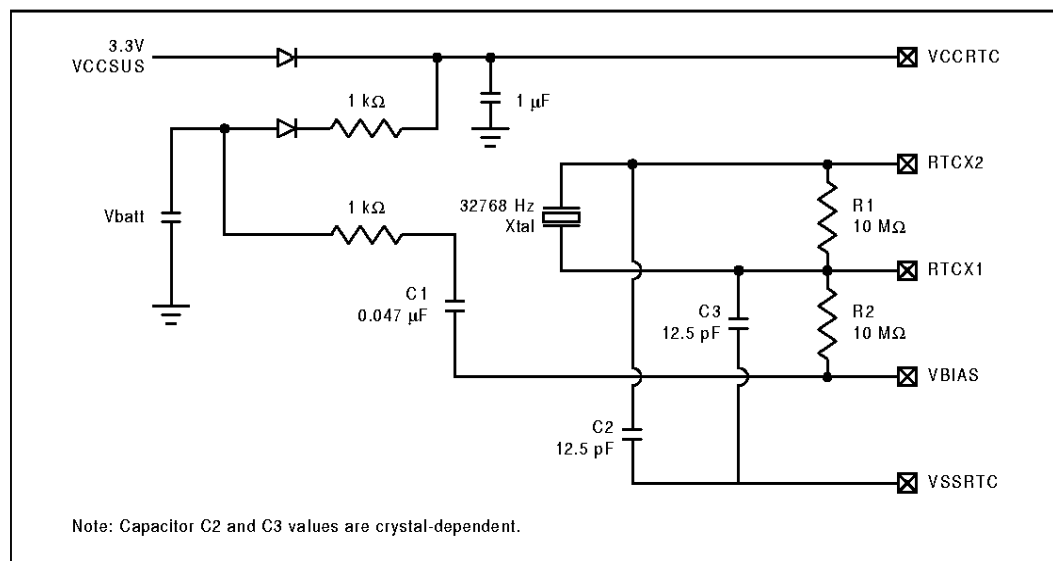
Table 2-19. Test Strap Definition

| Number of PCI Clocks RTCRST# Driven Low After PWROK Active | Test Mode                 |
|--|---------------------------|
| <4   | No Test Mode Invoked      |
| 4  | NAND Chain 1              |
| 5  | NAND Chain 2              |
| 6  | NAND Chain 3              |
| 7  | NAND Chain 4              |
| 8  | All "Z"                   |
| 9  | No Test Mode Invoked      |
| 10-24  | RESERVED - DO NOT ATTEMPT |
| >24  | No Test Mode Invoked      |

## 2.18.3 External RTC Circuitry

To reduce RTC well power consumption, the ICH implements a new internal oscillator circuit that is sensitive to step voltage changes in VccRTC and VBIAS. Figure 2-1 shows a schematic diagram of the circuitry required to condition these voltages to ensure correct operation of the ICH RTC

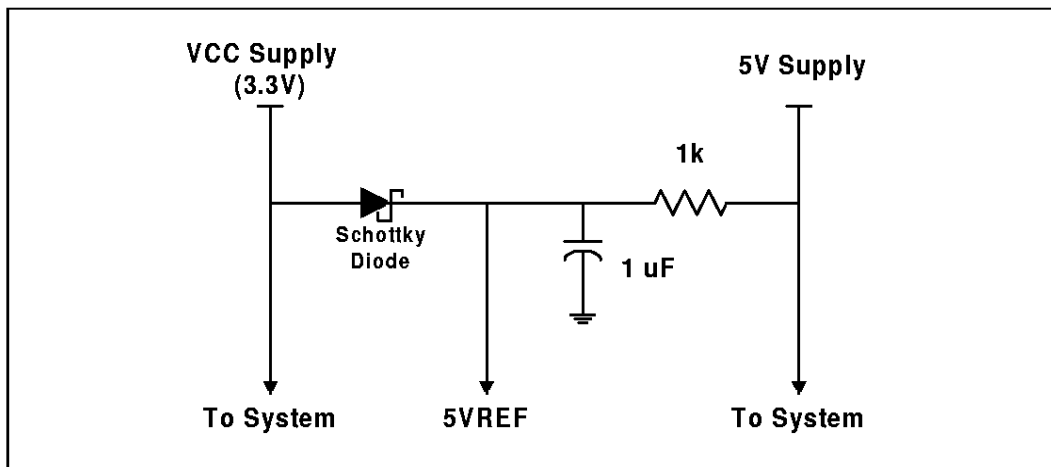
Figure 2-1. Required External RTC Circuit



### 2.18.4 5VREF / Vcc3\_3 Sequencing Requirements

The 5VREF pin is the reference voltage for 5V tolerance on inputs to the ICH. The 5VREF pin must power up before or simultaneous to Vcc3\_3, and must power down after or simultaneous to Vcc3\_3. Refer to Figure 2-2 for an example circuit schematic that may be used to ensure proper 5VREF sequencing.

Figure 2-2. Example 5VREF Sequencing Circuit





# Power Planes and Pin States

# 3

## 3.1 Power Planes

The ICH has five main power planes, as shown in Table 3-1 .

**Table 3-1. ICH Power Planes**

| Plane                     | Description   |
|---------------------------|---|
| Core<br>(3.3V)            | Powered by the main power supply. When the system is in the S3, S4, S5, or G3 state, this plane is assumed to be shut off.  |
| Hub Interface<br>(1.8V)   | Powered by the main power supply. When the system is in the S3, S4, S5, or G3 state, this plane is assumed to be shut off.  |
| Resume<br>(3.3V Stand-By) | Powered by the trickle power supply. In the G3 state, this plane is assumed to be shut off.   |
| RTC                       | When other power is available (from the main supply), external diode coupling will provide power to reduce the drain on the RTC battery. Operations from 3.3V down to 2.0V. |

## 3.2 Output and I/O Signal Planes and States

Table 3-2 shows the power plane associated with the output and I/O signals, as well as the state at various times. Within the table, the following terms are used:

|             |  |
|-------------|--|
| “High-Z”    | Tri-state. ICH not driving the signal high or low                          |
| “High”      | ICH is driving the signal to a logic ‘1’                                   |
| “Low”       | ICH is driving the signal to a logic ‘0’                                   |
| “Defined”   | Driven to a level that is defined by the function (will be high or low)    |
| “Undefined” | ICH is driving the signal, but the value is indeterminate                  |
| “Running”   | Clock is toggling or signal is transitioning because function not stopping |
| “Off”       | The power plane is off, so ICH is not driving                              |
| “?”         | Still needs to be defined.   |

Note that the signal levels are the same in S4 and S5

Table 3-2. Power Plane and States for Output and I/O Signals (Sheet 1 of 2)

| Signal Name                      | Power Plane | During PCIRST# | Immediate after PCIRST# | S1     | S3     | S4/S5  |
|----------------------------------|-------------|----------------|-------------------------|--------|--------|--------|
| <b>PCI Bus</b>                   |             |                |                         |        |        |        |
| AD[31:0]                         | Core        | High-Z         | Low                     | Low    | Off    | Off    |
| C/BE#[3:0]                       | Core        | High-Z         | High                    | High   | Off    | Off    |
| DEVSEL#                          | Core        | High-Z         | High-Z                  | High-Z | Off    | Off    |
| FRAME#                           | Core        | High-Z         | High-Z                  | High-Z | Off    | Off    |
| GNT[0:3]#                        | Core        | High           | High                    | High   | Off    | Off    |
| GNT[4:5]#<br>(ICH only: 82801AA) | Core        | High           | High                    | High   | Off    | Off    |
| GNT[A:B]#                        | Core        | High-Z         | High                    | High   | Off    | Off    |
| IRDY#, TRDY#                     | Core        | High-Z         | High-Z                  | High-Z | Off    | Off    |
| PAR                              | Core        | High-Z         | Low                     | Low    | Off    | Off    |
| PCIRST#                          | Resume      | Low            | High                    | High   | Low    | Low    |
| PERF#<br>(ICH only: 82801AA)     | Periphery   | High-Z         | High-Z                  | High-Z | Off    | Off    |
| PLOCK#                           | Core        | High-Z         | High-Z                  | High-Z | Off    | Off    |
| STOP#                            | Core        | High-Z         | High-Z                  | High-Z | Off    | Off    |
| <b>LPC Interface</b>             |             |                |                         |        |        |        |
| LAD[3:0]                         | Core        | High           | High                    | High   | Off    | Off    |
| LFRAME#                          | Core        | High           | High                    | High   | Off    | Off    |
| <b>IDE Interface</b>             |             |                |                         |        |        |        |
| PDA[2:0], SDA[2:0]               | Core        | Low            | Undefined               | Low    | Off    | Off    |
| PDCS1#, PDCS3#                   | Core        | High           | High                    | High   | Off    | Off    |
| PDD[15:0], SDD[15:0]             | Core        | High-Z         | High-Z                  | Off    | Off    | Off    |
| PDDACK#, DDACK#                  | Core        | High           | High                    | High   | Off    | Off    |
| PDIOR#, PDIOW#                   | Core        | High           | High                    | High   | Off    | Off    |
| SDCS1#, SDCS3#                   | Core        | High           | High                    | High   | Off    | Off    |
| SDIOR#, SDIOW#                   | Core        | High           | High                    | High   | Off    | Off    |
| <b>Interrupts</b>                |             |                |                         |        |        |        |
| PIRQ[A:D]#                       | Core        | High-Z         | High-Z                  | High-Z | Off    | Off    |
| SERIRQ                           | Core        | High-Z         | High-Z                  | High-Z | Off    | Off    |
| APICD[1:0]                       | Core        | High-Z         | High-Z                  | High-Z | Off    | Off    |
| <b>USB Interface</b>             |             |                |                         |        |        |        |
| USBP0P, USBP0N<br>USBP1P, USBP1N | Resume      | High-Z         | High-Z                  | High-Z | High-Z | High-Z |

Table 3-2. Power Plane and States for Output and I/O Signals (Sheet 2 of 2)

| Signal Name                                       | Power Plane | During PCIRST#                   | Immediate after PCIRST# | S1                    | S3      | S4/S5   |
|---|-------------|----------------------------------|-------------------------|-----------------------|---------|---------|
| <b>Power Management</b>                           |             |                                  |                         |                       |         |         |
| SLP_S3#   | Resume      | High                             | High                    | High                  | Low     | Low     |
| SLP_S5#   | Resume      | High                             | High                    | High                  | High    | Low     |
| SUS_STAT#   | Resume      | High                             | High                    | High                  | Low     | Low     |
| SUSCLK<br>(ICH only: 82801AA)                     | Resume      | Running                          |                         |                       |         |         |
| <b>Processor Interface</b>                        |             |                                  |                         |                       |         |         |
| A20M#   | Core        | Note 1                           | High-Z                  | High-Z                | Off     | Off     |
| IGNNE#  | Core        | Note 1                           | High-Z                  | High-Z                | Off     | Off     |
| INIT#   | Core        | High-Z                           | High-Z                  | High-Z                | Off     | Off     |
| INTR  | Core        | Note 1                           | Low                     | Low                   | Off     | Off     |
| NMI   | Core        | Note 1                           | Low                     | Low                   | Off     | Off     |
| SMI#  | Core        | High-Z                           | High-Z                  | High-Z                | Off     | Off     |
| STPCLK#   | Core        | High-Z                           | High-Z                  | Low                   | Off     | Off     |
| <b>SMBus Interface</b>                            |             |                                  |                         |                       |         |         |
| SMBCLK, SMBDATA                                   | Resume      | High-Z                           | High-Z                  | High-Z                | High-Z  | High-Z  |
| <b>Miscellaneous Signals</b>                      |             |                                  |                         |                       |         |         |
| SPKR  | Core        | Tri-stated with internal pull-up | Low                     | Low                   | Off     | Off     |
| <b>AC'97 Interface</b>                            |             |                                  |                         |                       |         |         |
| AC_RST#   | Resume      | Low                              | Low                     | Cold Reset Bit (High) | Low     | Low     |
| AC_SDOUT  | Core        | Low                              | Running                 | Low                   | Low     | Off     |
| AC_SYNC   | Core        | Low                              | Running                 | Low                   | Off     | Off     |
| <b>Unmuxed GPIO Signals</b>                       |             |                                  |                         |                       |         |         |
| GPIO[21]  | Core        | High                             | High                    | Defined               | Off     | Off     |
| GPIO[22]  | Core        | Low                              | Low                     | Defined               | Off     | Off     |
| GPIO[23]  | Core        | Low                              | Low                     | Defined               | Off     | Off     |
| GPIO[26]<br>(unmuxed in ICH0 only:<br>82801AB)    | Resume      | Low                              | Low                     | Defined               | Defined | Defined |
| GPIO[27:28]<br>(unmuxed in ICH0 only:<br>82801AB) | Resume      | Low                              | Low                     | Defined               | Defined | Defined |

**Note:** 1. ICH sets these signals at reset for processor frequency strap.

### 3.3 Power Planes for Input Signals

Table 3-3 shows the power plane associated with each input signal, as well as what device drives the signal at various times. Valid states include:

High

Low

Static: Will be high or low, but will not change

Driven: Will be high or low, and is allowed to change

Running: For input clocks

Table 3-3. Power Plane for Input Signals

| Signal Name                      | Power Well    | Driver During Reset      | S1      | S3      | S5      |
|----------------------------------|---------------|--------------------------|---------|---------|---------|
| A20GATE                          | Core          | External Microcontroller | Static  | Low     | Low     |
| AC_BIT_CLK                       | Core          | AC'97 Codec              | Low     | Low     | Low     |
| AC_SDIN[1:0]                     | Resume        | AC'97 Codec              | Low     | Low     | Low     |
| APICCLK                          | Core          | Clock Generator          | Running | Low     | Low     |
| CLK14                            | Core          | Clock Generator          | Running | Low     | Low     |
| CLK48                            | Core          | Clock Generator          | Running | Low     | Low     |
| FERR#                            | Core          | Processor                | Static  | Low     | Low     |
| CLK66                            | Hub Interface | Clock Generator          | Running | Low     | Low     |
| INTRUDER#                        | RTC           | External Switch          | Driven  | Driven  | Driven  |
| IRQ[15:14]                       | Core          | IDE                      | Driven  | Low     | Low     |
| LDRQ[0]#                         | Core          | LPC Devices              | High    | Low     | Low     |
| LDRQ[1]#                         | Resume        | LPC Devices              | High    | Low     | Low     |
| OC[1:0]#                         | Resume        | External Pull-Ups        | Defined | Defined | Defined |
| PCICLK                           | Core          | Clock Generator          | Running | Low     | Low     |
| PDDREQ                           | Core          | IDE Device               | Static  | Low     | Low     |
| PIORDY                           | Core          | IDE Device               | Static  | Low     | Low     |
| PME#                             | Resume        | Internal Pull-Up         | Driven  | Driven  | Driven  |
| PWRBTN#                          | Resume        | Internal Pull-Up         | Driven  | Driven  | Driven  |
| PWROK                            | Core          | System Power Supply      | Driven  | Low     | Low     |
| RCIN#                            | Core          | External Microcontroller | High    | Low     | Low     |
| REQ[0:3]#                        | Core          | PCI Master               | Driven  | Low     | Low     |
| REQ[4:5]#<br>(ICH only: 82801AA) | Core          | PCI Master               | Driven  | Low     | Low     |
| REQ[B:A]#                        | Core          | PC/PCI Devices           | Driven  | Low     | Low     |
| RI#                              | Resume        | Serial Port Buffer       | Driven  | Defined | Defined |
| RSMRST#                          | RTC           | External RC circuit      | High    | High    | High    |
| RTCST#                           | RTC           | External RC circuit      | High    | High    | High    |
| SDDREQ                           | Core          | IDE Drive                | Static  | Low     | Low     |
| SERR#                            | Core          | PCI Bus Peripherals      | High    | Low     | Low     |
| SIORDY                           | Core          | IDE Drive                | Static  | Low     | Low     |
| THRM#                            | Core          | Thermal Sensor           | Driven  | Low     | Low     |

### 3.4 Integrated Pull-Ups and Pull-Downs

Table 3-4 shows the ICH signals that have integrated pull-up or pull-down resistors.

**Table 3-4. Integrated Pull-Ups and Pull-Downs**

| Signal                            | Resistor Value | Resistor Type |
|-----------------------------------|----------------|---------------|
| PME#                              | 9K $\pm$ 3K    | pull-up       |
| PWRBTN#                           | 9K $\pm$ 3K    | pull-up       |
| LAD[3:0]# / FWH[3:0]#             | 9K $\pm$ 3K    | pull-up       |
| SPKR <sup>1</sup>                 | 22K $\pm$ 4K   | pull-up       |
| AC_SDOUT <sup>1</sup>             | 22K $\pm$ 4K   | pull-down     |
| AC_SDIN[1] / GPIO[9] <sup>2</sup> | 22K $\pm$ 4K   | pull-down     |
| AC_SDIN[0] <sup>2</sup>           | 22K $\pm$ 4K   | pull-down     |
| AC_BITCLK <sup>2</sup>            | 22K $\pm$ 4K   | pull-down     |

**NOTES:**

1. The pull-up or pull-down on this signal is only enabled at boot/reset for strapping function.
2. The pull-down resistor on this signal is only enabled when the ACLINK Shut Off bit in the AC'97 Global Control Register (in AC'97 I/O space) is set to 1. At all other times, the pull-down resistor is disabled.

### 3.5 IDE Integrated Series Termination Resistors

Table 3-5 shows the ICH IDE signals that have integrated series termination resistors.

**Table 3-5. IDE Series Termination Resistors**

| Signal   | Integrated Series Termination Resistor Value | Notes |
|--|--|-------|
| PDD[15:0], SDD[15:0]   | approximately 33 ohms                        | 1     |
| PDIO# <sup>1</sup> , SDIO# <sup>1</sup> , PDIO# <sup>1</sup> , PDIO# <sup>1</sup> , PDREQ, SDREQ, PDDACK#, SDDACK#, PIORDY, SIORDY, PDA[2:0], SDA[2:0], PDCS1#, SDCS1#, PDCS3#, SDCS3#, IRQ14, IRQ15 | approximately 22 ohms                        | 2     |

**NOTES:**

1. Simulation data indicates that these integrated series termination resistors are a nominal 33 ohms; they can range from 28 ohms to 46 ohms.
2. Simulation data indicates that these integrated series termination resistors are a nominal 22 ohms; they can range from 19 ohms to 29 ohms.



# Clock Domains

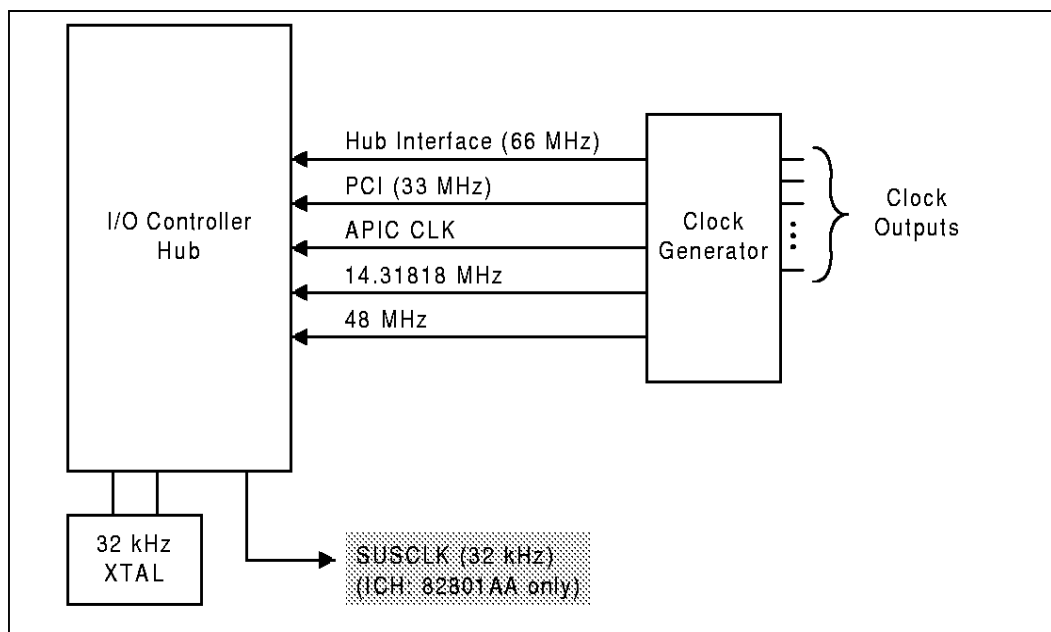
# 4

Table 4-1 shows the system clock domains. Figure 4-1 shows the assumed connection of the various system components, including the clock generator. For complete details of the system clocking solution refer to the system's clock generator component specification.

**Table 4-1. ICH and System Clock Domains**

| Clock Domain | Frequency    | Source               | Usage   |
|--------------|--------------|----------------------|---|
| ICH CLK66    | 66 MHz       | Main Clock Generator | Hub Interface. Shut off during S3 or below.   |
| ICH PCICLK   | 33 MHz       | Main Clock Generator | PCI Clock. This clock remains on during S0 state, but can be shut off during S3 or below. |
| System PCI   | 33 MHz       | Main Clock Generator | PCI Bus, LPC interface. These only go to external PCI and LPC devices.                    |
| ICH USB      | 48 MHz       | Main Clock Generator | USB Controller.   |
| ICH CLK14    | 14.31818 MHz | Main Clock Generator | Used for ACPI timer   |
| ICH AC'97    | 12.288 MHz   | AC'97 Codec          | AC'97 Link. Generated by AC'97 CODEC  |
| RTC          | 32.768 kHz   | ICH                  | RTC, Power Management. ICH has its own oscillator.  |
| ICH APIC     | 16.667 MHz   | Main Clock Generator | Used for ICH-CPU interrupt messages.  |

**Figure 4-1. Conceptual System Clock Diagram**







# Functional Description

# 5

## 5.1 Hub Interface to PCI Bridge (D30:F0)

The hub interface to PCI Bridge resides in PCI Device 30, Function 0 on bus #0. This portion of the ICH implements the buffering and control logic between PCI and the Hub Interface. The arbitration for the PCI bus is handled by this PCI device. The PCI decoder in this device must decode the ranges for the Hub Interface. All register contents will be lost when core well power is removed.

### 5.1.1 PCI Bus Interface

The ICH PCI interface provides a 33 MHz, Rev. 2.2 compliant implementation. All PCI signals are 5V tolerant, except PME#. The ICH integrates a PCI arbiter that supports up to four (ICH0: 82801AB) or six (ICH: 82801AA) external PCI bus masters in addition to the internal ICH requests.

**Note:** Most transactions targeted to the ICH will first appear on the external PCI bus before being claimed back by the ICH. The exceptions are I/O cycles involving USB, IDE, and AC'97. These transactions will complete over the Hub Interface without appearing on the external PCI bus. *Configuration cycles* targeting USB, IDE or AC'97 will appear on the PCI bus.

**Note:** If the ICH is programmed for positive decode, the ICH will grab the cycles appearing on the external PCI bus in medium decode time. If the ICH is programmed for subtractive decode, the ICH will grab these cycles in subtractive time. If the ICH is programmed for subtractive decode, these cycles can be grabbed by another positive decode agent out on PCI.

**Note:** This architecture enables the ability to boot off of a PCI card that positively decodes the boot cycles. To boot off a PCI card it is necessary to keep the ICH in subtractive decode mode. When booting off a PCI card, the BOOT\_STS bit (bit 2, TCO2 Status Register) will be set.

### 5.1.2 PCI-to-PCI Bridge Model

From a software perspective, the ICH contains a PCI-to-PCI bridge. This bridge connects the hub interface to the PCI bus. By using the PCI-to-PCI bridge software model, the ICH can have its decode ranges programmed by existing plug-and-play software such that PCI ranges do not conflict with graphics ranges in the host controller.

### 5.1.3 IDSEL to Device Number Mapping

When addressing devices on the external PCI bus (with the PCI slots) the ICH asserts one address signal as an IDSEL. When accessing device 0, the ICH asserts AD16. When accessing Device 1, the ICH asserts AD17. This mapping continues all the way up to device 15 where the ICH asserts AD31. Note that the ICH's internal functions (AC'97, IDE, USB, and PCI Bridge) are enumerated like they are on a separate PCI bus (the Hub Interface) from the external PCI bus.

### 5.1.4 SERR#/PERR#/NMI# Functionality

Figure 5-1, Figure 5-2, and Figure 5-3 show the logic used to generate/report SERR#, PERR#, and NMI#. Note that PERR# is only on the ICH (82801AA).

There are several internal and external sources that can cause SERR#. The ICH can be programmed to cause an NMI based on detecting that an SERR# condition has occurred. The NMI can also be routed to instead cause an SMI#. Note that the ICH does not drive the external PCI bus SERR# signal active onto the PCI bus. The external SERR# signal is an input into the ICH driven only by external PCI devices.

The ICH (82801AA) can detect and report different parity errors in the system. The ICH can be programmed to cause an NMI (or SMI# if NMI is routed to SMI#) based on detecting a parity error.

Figure 5-1. NMI# Generation Logic

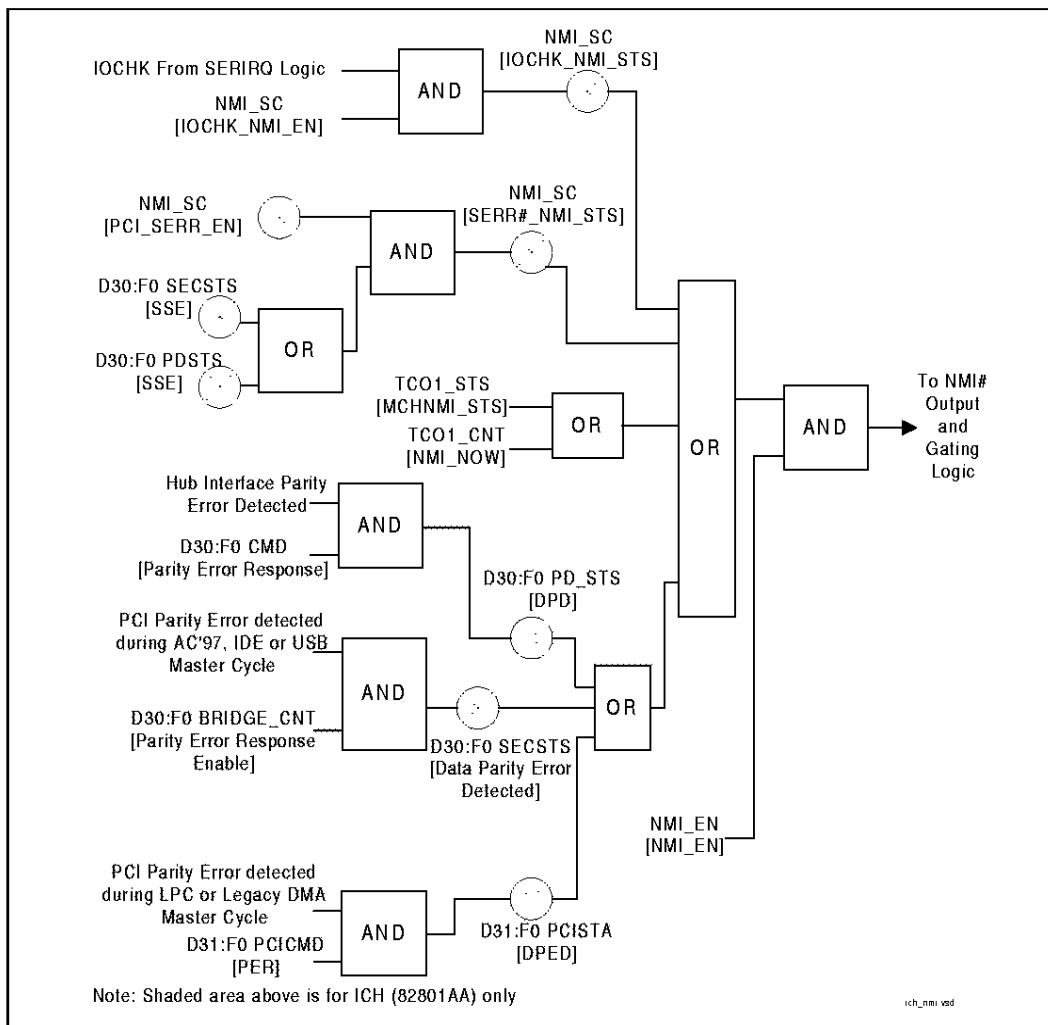


Figure 5-2. Primary Device Status Register Error Reporting Logic

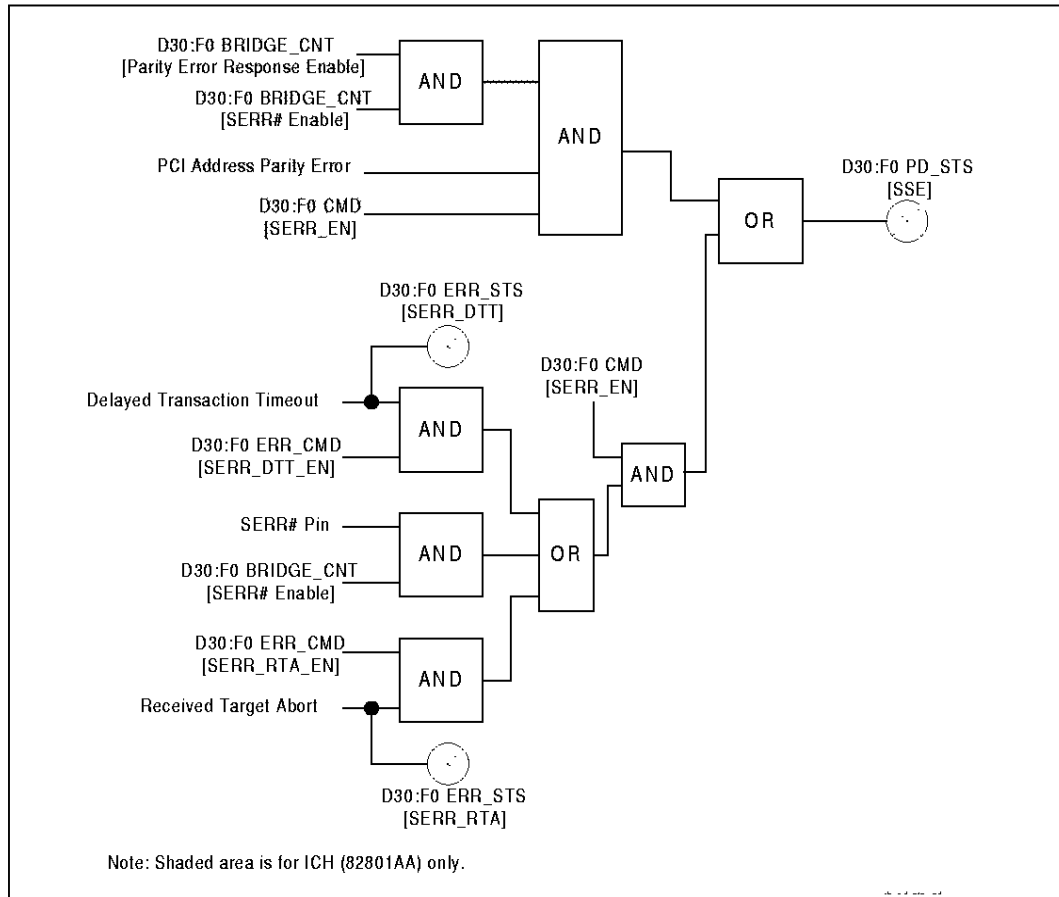
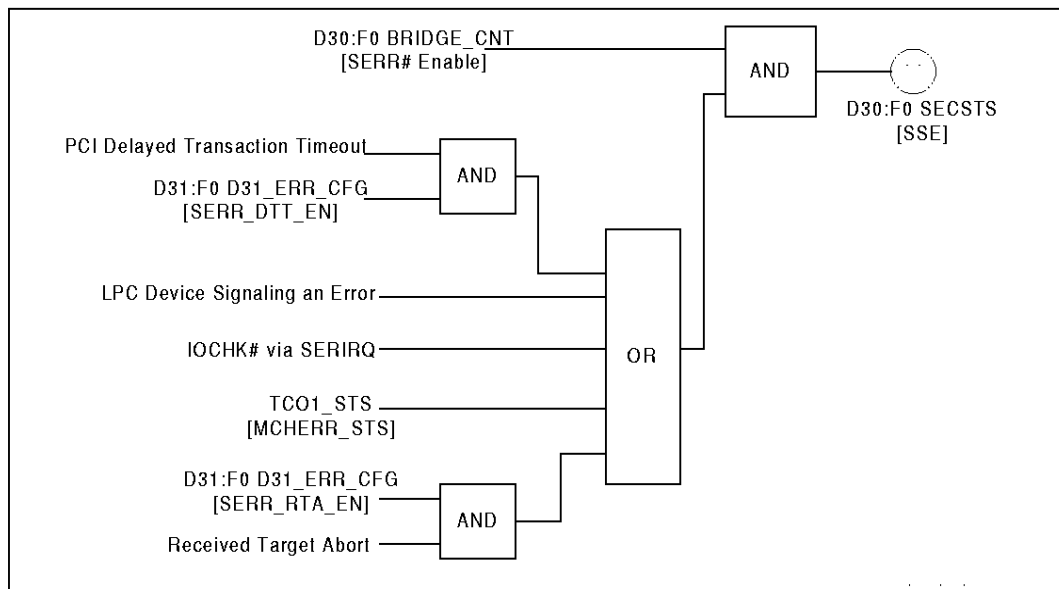


Figure 5-3. Secondary Status Register Error Reporting Logic



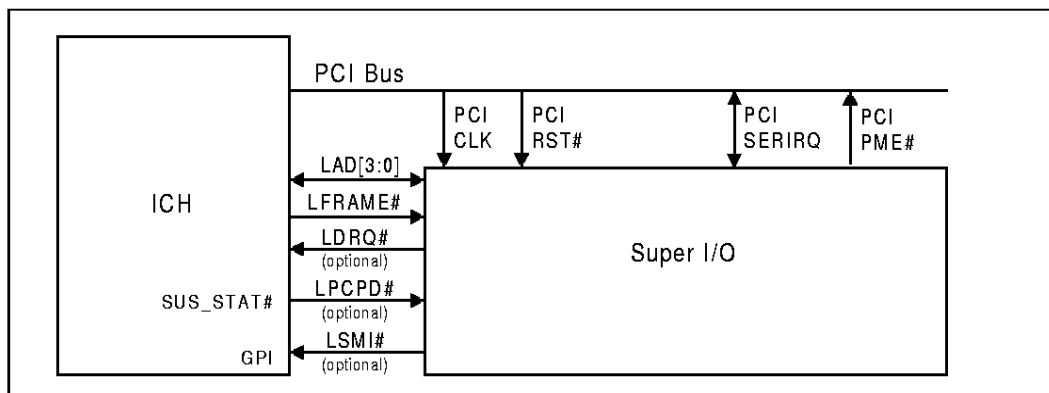
## 5.2 LPC Bridge (with System & Management Functions) (D31:F0)

The Low Pin Count (LPC) Bridge function of the ICH resides in PCI Device 31:Function 0. In addition to the LPC bridge function, D31:F0 contains other functional units including DMA, Interrupt Controllers, Timers, Power Management, System Management, GPIO, and RTC. In this chapter, registers and functions associated with other functional units (power management, GPIO, USB, IDE, etc.) are described in their respective sections.

### 5.2.1 LPC Interface

The ICH implements an LPC Interface as described in the LPC 1.0 specification. The LPC Interface to the ICH is shown in Figure 5-4. Note that the ICH implements all of the signals that are shown as optional, but peripherals are not required to do so.

Figure 5-4. LPC Interface Diagram



#### 5.2.1.1 LPC Cycle Types

The ICH implements all of the cycle types described in the LPC Interface 1.0 specification. Table 5-1 shows the cycle types supported by the ICH.

Table 5-1. LPC Cycle Types Supported

| Cycle Type       | Comment  |
|------------------|--|
| Memory Read      | Single: 1 byte only  |
| Memory Write     | Single: 1 byte only  |
| I/O Read         | 1 byte only. ICH breaks up 16 and 32-bit processor cycles into multiple 8-bit transfers. |
| I/O Write        | 1 byte only. ICH breaks up 16 and 32-bit processor cycles into multiple 8-bit transfers. |
| DMA Read         | Can be 1, or 2 bytes   |
| DMA Write        | Can be 1, or 2 bytes   |
| Bus Master Read  | Can be 1, 2, or 4 bytes. (See Note below)  |
| Bus Master Write | Can be 1, 2, or 4 bytes. (See Note below)  |

**NOTE:** Bus Master Read or Write cycles must be naturally aligned. For example, a 1-byte transfer can be to any address. However, the 2-byte transfer must be word aligned (i.e., with an address where A0=0). A dword transfer must be dword aligned (i.e., with an address where A1 and A0 are both 0)

### 5.2.1.2 Start Field Definition

Table 5-2. Start Field Bit Definitions

| Bits[3:0]<br>Encoding | Definition                               |
|-----------------------|--|
| 0000                  | Start of cycle for a generic target.     |
| 0010                  | Grant for bus master 0.                  |
| 0011                  | Grant for bus master 1.                  |
| 1111                  | Stop/Abort: End of a cycle for a target. |

NOTE: All other encodings are RESERVED.

### 5.2.1.3 Cycle Type / Direction (CYCTYPE + DIR)

The ICH always drives bit 0 of this field to 0. Peripherals running bus master cycles must also drive bit 0 to 0. Table 5-3 shows the valid bit encodings:

Table 5-3. Cycle Type Bit Definitions

| Bits[3:2] | Bit[1] | Definition  |
|-----------|--------|---|
| 00        | 0      | I/O Read  |
| 00        | 1      | I/O Write   |
| 01        | 0      | Memory Read   |
| 01        | 1      | Memory Write  |
| 10        | 0      | DMA Read  |
| 10        | 1      | DMA Write   |
| 11        | x      | Reserved. If a peripheral performing a bus master cycle generates this value, the ICH will abort the cycle. |

### 5.2.1.4 SIZE

Bits[3:2] are reserved. The ICH always drives them to 00. Peripherals running bus master cycles are also supposed to drive 00 for bits 3:2; however, the ICH ignores those bits. Bits[1:0] are encoded as shown in Table 5-4.

Table 5-4. Transfer Size Bit Definition

| Bits[1:0] | Size   |
|-----------|--|
| 00        | 8 bit transfer (1 byte)  |
| 01        | 16-bit transfer (2 bytes)  |
| 10        | Reserved. The ICH never drives this combination. If a peripheral running a bus master cycle drives this combination, the ICH may abort the transfer. |
| 11        | 32 bit transfer (4 bytes)  |

### 5.2.1.5 SYNC

Valid values for the SYNC field are shown in Table 5-5.

**Table 5-5. SYNC Bit Definition**

| Bits[3:0] | indication  |
|-----------|---|
| 0000      | <b>Ready:</b> SYNC achieved with no error. For DMA transfers, this also indicates DMA request de-assertion and no more transfers desired for that channel.  |
| 0101      | <b>Short Wait:</b> Part indicating wait states. For bus master cycles, the ICH does not use this encoding. It, instead, uses the Long Wait encoding (see next encoding below).  |
| 0110      | <b>Long Wait:</b> Part indicating wait states, and many wait states will be added. This encoding driven by the ICH for bus master cycles, rather than the Short Wait (0101).  |
| 1001      | <b>Ready More (Used only by peripheral for DMA cycle):</b> SYNC achieved with no error and more DMA transfers desired to continue after this transfer. This value is valid only on DMA transfers and is not allowed for any other type of cycle.  |
| 1010      | <b>Error:</b> Sync achieved with error. This is generally used to replace the SERR# or IOCHK# signal on the PCI/ISA bus. It indicates that the data is to be transferred, but there is a serious error in this transfer. For DMA transfers, this not only indicates an error, but also indicates DMA request de-assertion and no more transfers desired for that channel. |

**NOTE:** All other combinations are RESERVED.

### 5.2.1.6 SYNC Time-out

There are several error cases that can occur on the LPC Interface. Table 5-6 indicates the failing case and the ICH response.

**Table 5-6. ICH Response to Sync Failures**

| Possible Sync Failure   | ICH Response  |
|---|---|
| ICH starts a Memory, I/O, or DMA cycle, but no device drives a valid SYNC after 4 consecutive clocks. This could occur if the processor tries to access an I/O location to which no device is mapped.                               | ICH aborts the cycle after the 4 <sup>th</sup> clock.     |
| ICH drives a Memory, I/O, or DMA cycle, and a peripheral drives more than 8 consecutive valid SYNC to insert wait states using the Short ('0101b') encoding for SYNC. This could occur if the peripheral is not operating properly. | Continues waiting   |
| ICH starts a Memory, I/O, or DMA cycle, and a peripheral drives an invalid SYNC pattern. This could occur if the peripheral is not operating properly or if there is excessive noise on the LPC Interface.                          | ICH aborts the cycle when the invalid Sync is recognized. |

**NOTE:** There may be other peripheral failure conditions; however, these are not handled by the ICH.

### 5.2.1.7 SYNC Error Indication

The SYNC protocol allows the peripheral to report an error via the LAD[3:0] = '1010b' encoding. The intent of this encoding is to give peripherals a method of communicating errors to aid higher layers with more robust error recovery.

If the ICH was reading data from a peripheral, data is still transferred in the next two nibbles. This data may be invalid, but it must be transferred by the peripheral. If the ICH was writing data to the peripheral, the data had already been transferred.

In the case of multiple byte cycles, such as for memory and DMA cycles, an error SYNC terminates the cycle. Therefore, if the ICH is transferring 4 bytes from a device, if the device returns the error SYNC in the first byte, the other three bytes are not transferred.

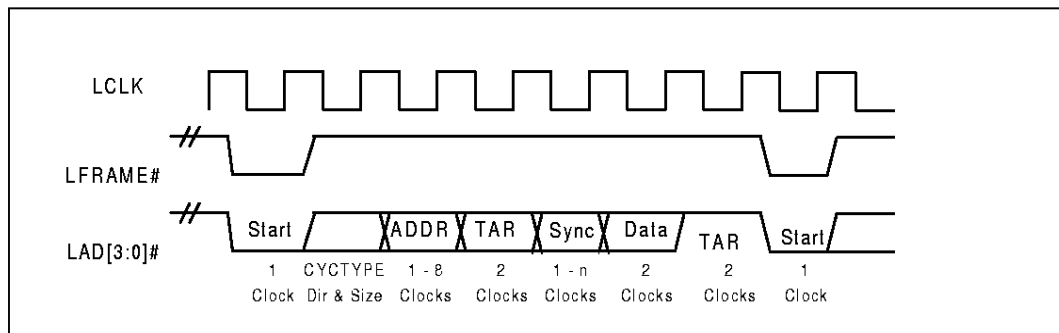
Upon recognizing the SYNC field indicating an error, the ICH treats this the same as IOCHK# going active on the ISA bus.

### 5.2.1.8 LFRAME# Usage

#### Start of Cycle

For Memory, I/O, and DMA cycles, the ICH asserts LFRAME# for 1 clock at the beginning of the cycle (Figure 5-5) During that clock, the ICH drives LAD[3:0] with the proper START field.

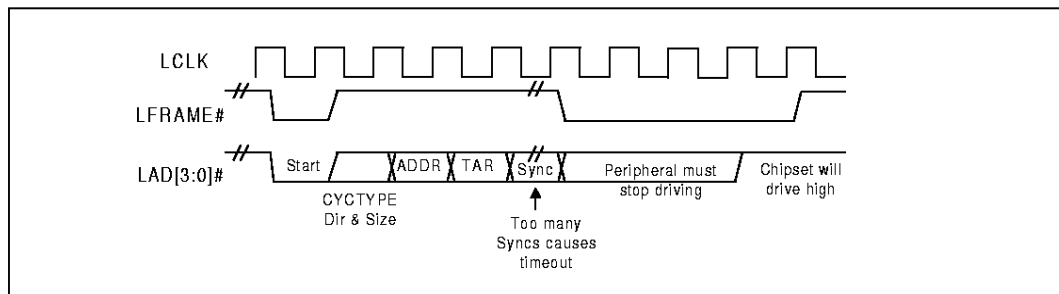
Figure 5-5. Typical Timing for LFRAME#



#### Abort Mechanism

When performing an Abort, the ICH drives LFRAME# active for 4 consecutive clocks. On the 4<sup>th</sup> clock, it drives LAD[3:0] to '1111b'.

Figure 5-6. Abort Mechanism



The ICH performs an abort for the following cases (possible failure cases):

- ICH starts a Memory, I/O, or DMA cycle, but no device drives a valid SYNC after 4 consecutive clocks.
- ICH starts a Memory, I/O, or DMA cycle, and the peripheral drives an invalid SYNC pattern.
- A peripheral drives an illegal address when performing bus master cycles.
- A peripheral drives an invalid value.

### 5.2.1.9 I/O Cycles

For I/O cycles targeting registers specified in the ICH's decode ranges, the ICH performs I/O cycles as defined in the LPC specification. These are 8-bit transfers. If the processor attempts a 16-bit or 32-bit transfer, the ICH breaks it up into multiple 8-bit transfers.

**Note:** If the cycle is not claimed by any peripheral (and subsequently aborted), the ICH returns a value of all 1's (FFh) to the processor. This is to maintain compatibility with ISA I/O cycles where pull-up resistors would keep the bus high if no device responds.

### 5.2.1.10 Bus Master Cycles

The ICH supports Bus Master cycles and requests (using LDRQ#) as defined in the LPC specification. The ICH has two LDRQ# inputs, and thus supports two separate bus master devices. It uses the associated START fields for Bus Master 0 ('0010b') or Bus Master 1 ('0011b').

### 5.2.1.11 Configuration and ICH Implications

#### LPC Interface Decoders

To allow the I/O cycles and memory mapped cycles to go to the LPC Interface, the ICH includes several decoders. During configuration, the ICH must be programmed with the same decode ranges as the peripheral. The decoders are programmed via the Device 31:Function 0 configuration space.

#### Bus Master Device Mapping and START Fields

Bus Masters must have a unique START field. In the case of the ICH, which supports 2 bus masters, it drives 0010 for the START field for grants to bus master #0 (requested via LDRQ[0]#) and 0011 for grants to bus master #1 (requested via LDRQ[1]#). Thus no registers are needed to config the START fields for a particular bus master.

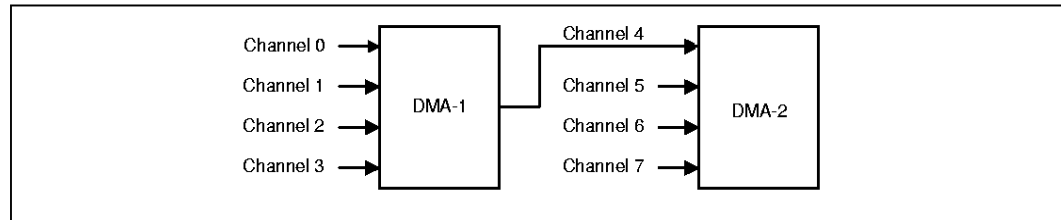
## 5.3 DMA Operation (D31:F0)

The ICH supports two types of DMA: LPC, and PC/PCI. DMA via LPC is similar to ISA DMA. LPC DMA and PC/PCI DMA use the ICH's DMA controller.

The DMA controller has registers that are fixed in the lower 64 KB of I/O space. For PCI configuration, the controller is configured using registers in the PCI configuration space. These registers allow configuration of individual channels for use by LPC or PC/PCI DMA.

The DMA circuitry incorporates the functionality of two 82C37 DMA controllers with seven independently programmable channels (Figure 5-7). DMA Controller 1 (DMA-1) corresponds to DMA Channels 0-3 and DMA Controller 2 (DMA-2) corresponds to Channels 5-7. DMA Channel 4 is used to cascade the two controllers and defaults to cascade mode in the DMA Channel Mode (DCM) Register. Channel 4 is not available for any other purpose. In addition to accepting requests from DMA slaves, the DMA controller also responds to requests that software initiates. Software may initiate a DMA service request by setting any bit in the DMA Channel Request Register to a 1.



**Figure 5-7. ICH DMA Controller**


Each DMA channel is hardwired to the compatible settings for DMA device size: channels [3:0] are hardwired to 8-bit, count-by-bytes transfers, and channels [7:5] are hardwired to 16-bit, count-by-words (address shifted) transfers.

The ICH provides 24-bit addressing in compliance with the ISA-Compatible specification. Each channel includes a 16-bit ISA-Compatible Current Register which holds the 16 least-significant bits of the 24-bit address, an ISA-Compatible Page Register which contains the eight next most significant bits of address.

The DMA controller also features refresh address generation, and autoinitialization following a DMA termination.

### 5.3.1 Channel Priority

For priority resolution, the DMA consists of two logical channel groups: channels 0–3 and channels 4–7. Each group may be in either fixed or rotate mode, as determined by the DMA Command Register.

DMA I/O slaves normally assert their DREQ line to arbitrate for DMA service. However, a software request for DMA service can be presented through each channel's DMA Request Register. A software request is subject to the same prioritization as any hardware request. Please see the detailed register description for Request Register programming information in the DMA Register description section.

#### Fixed Priority

The initial fixed priority structure is as follows:

|                                |           |
|--------------------------------|-----------|
| High priority.....Low priority |           |
| (0, 1, 2, 3)                   | (5, 6, 7) |

The fixed priority ordering is 0, 1, 2, 3, 5, 6, and 7. In this scheme, Channel 0 has the highest priority, and channel 7 has the lowest priority. Channels [3:0] of DMA-1 assume the priority position of Channel 4 in DMA-2, thus taking priority over channels 5, 6, and 7.

#### Rotating Priority

Rotation allows for "fairness" in priority resolution. The priority chain rotates so that the last channel serviced is assigned the lowest priority in the channel group (0–3, 5–7). Channels 0–3 rotate as a group of 4. They are always placed between Channel 5 and Channel 7 in the priority list. Channel 5–7 rotate as part of a group of 4. That is, channels (5–7) form the first three positions in the rotation, while channel group (0–3) comprises the fourth position in the arbitration.

### 5.3.2 Address Compatibility Mode

Whenever the DMA is operating, the addresses do not increment or decrement through the High and Low Page Registers. Therefore, if a 24-bit address is 01FFFFh and increments, the next address is 010000h, not 020000h. Similarly, if a 24-bit address is 020000h and decrements, the next address is 02FFFFh, not 01FFFFh. This is compatible with the 82C37 and Page Register implementation used in the PC-AT. This mode is set after CPURST is valid.

### 5.3.3 Summary of DMA Transfer Sizes

Table 5-7 lists each of the DMA device transfer sizes. The column labeled "Current Byte/Word Count Register" indicates that the register contents represents either the number of bytes to transfer or the number of 16-bit words to transfer. The column labeled "Current Address Increment/Decrement" indicates the number added to or taken from the Current Address register after each DMA transfer cycle. The DMA Channel Mode Register determines if the Current Address Register is incremented or decremented.

#### Address Shifting When Programmed for 16-Bit I/O Count by Words

Table 5-7. DMA Transfer Size

| DMA Device Data Size And Word Count          | Current Byte/Word Count Register | Current Address Increment/Decrement |
|--|----------------------------------|-------------------------------------|
| 8-Bit I/O, Count By Bytes                    | Bytes                            | 1                                   |
| 16-Bit I/O, Count By Words (Address Shifted) | Words                            | 1                                   |

The ICH maintains compatibility with the implementation of the DMA in the PC AT which used the 82C37. The DMA shifts the addresses for transfers to/from a 16-bit device count-by-words. Note that the least significant bit of the Low Page Register is dropped in 16-bit shifted mode. When programming the Current Address Register (when the DMA channel is in this mode), the Current Address must be programmed to an even address with the address value shifted right by one bit. The address shifting is shown in Table 5-8.

Table 5-8. Address Shifting in 16-bit I/O DMA Transfers

| Output Address            | 8-Bit I/O Programmed Address (Ch 0-3) | 16-Bit I/O Programmed Address (Ch 5-7) (Shifted) |
|---------------------------|---------------------------------------|--|
| A0<br>A[16:1]<br>A[23:17] | A0<br>A[16:1]<br>A[23:17]             | 0<br>A[15:0]<br>A[23:17]                         |

**NOTE:** The least significant bit of the Page Register is dropped in 16-bit shifted mode.

### 5.3.4 Autoinitialize

By programming a bit in the DMA Channel Mode Register, a channel may be set up as an autoinitialize channel. When a channel undergoes autoinitialization, the original values of the Current Page, Current Address and Current Byte/Word Count Registers are automatically restored from the Base Page, Address, and Byte/Word Count Registers of that channel following TC. The Base Registers are loaded simultaneously with the Current Registers by the microprocessor when the DMA channel is programmed and remain unchanged throughout the DMA service. The mask bit is not set when the channel is in autoinitialize. Following autoinitialize, the channel is ready to perform another DMA service, without processor intervention, as soon as a valid DREQ is detected.

### 5.3.5 Software Commands

There are three additional special software commands that the DMA controller can execute. The three software commands are:

1. Clear Byte Pointer Flip-Flop
2. Master Clear
3. Clear Mask Register

They do not depend on any specific bit pattern on the data bus.

#### Clear Byte Pointer Flip-Flop

This command is executed prior to writing or reading new address or word count information to/from the DMA controller. This initializes the flip-flop to a known state so that subsequent accesses to register contents by the microprocessor addresses upper and lower bytes in the correct sequence.

When the Host processor is reading or writing DMA registers, two Byte Pointer flip-flops are used; one for channels 0–3 and one for channels 4–7. Both of these act independently. There are separate software commands for clearing each of them (0Ch for channels 0–3, 0D8h for channels 4–7).

#### DMA Master Clear

This software instruction has the same effect as the hardware reset. The Command, Status, Request, and Internal First/Last Flip-Flop Registers are cleared and the Mask Register is set. The DMA controller enters the idle cycle.

There are two independent master clear commands:

- 0Dh which acts on channels 0–3
- 0DAh which acts on channels 4–7.

#### Clear Mask Register

This command clears the mask bits of all four channels, enabling them to accept DMA requests. I/O port 00Eh is used for channels 0–3 and I/O port 0DCh is used for channels 4–7.

## 5.4 PCI DMA

The ICH provides support for the PC/PCI DMA protocol. PC/PCI DMA uses dedicated REQUEST and GRANT signals to permit PCI devices to request transfers associated with specific DMA channels. Upon receiving a request and getting control of the PCI bus, ICH performs a two-cycle transfer. For example, if data is to be moved from the peripheral to main memory, ICH first reads data from the peripheral and then write it to main memory. The location in main memory is the Current Address Registers in the 8237. ICH supports up to 2 PC/PCI REQ/GNT pairs, REQ[A:B]# and GNT[A:B]#.

A 16-bit register is included in the ICH Function 0 configuration space at offset 90h. It is divided into seven 2-bit fields that are used to configure the 7 DMA channels. Each DMA channel can be configured to one of two options:

- LPC DMA
- PC/PCI style DMA using the REQ/GNT signals

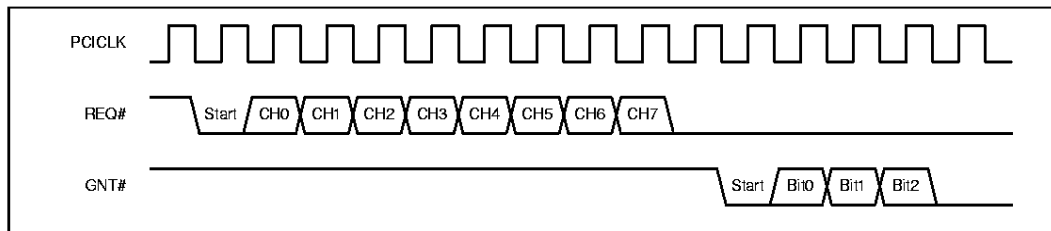
It is not possible for a particular DMA channel to be configured for more than one style of DMA; however, the seven channels can be programmed independently. For example, channel 3 could be set up for PC/PCI and channel 5 set up for LPC DMA.

The ICH REQ[A:B]# and GNT[A:B]# can be configured for support of a PC/PCI DMA Expansion agent. The PCI DMA Expansion agent can then provide DMA service or ISA Bus Master service using the ICH DMA controller. The REQ#/GNT# pair must follow the PC/PCI serial protocol described below.

### 5.4.1 PCI DMA Expansion Protocol

The PCI expansion agent must support the PCI expansion Channel Passing Protocol defined in Figure 5-8 for both the REQ# and GNT# pins.

Figure 5-8. DMA Serial Channel Passing Protocol



The requesting device must encode the channel request information as shown above, where CH0–CH7 are one clock active high states representing DMA channel requests 0–7.

The ICH encodes the granted channel on the GNT# line as shown above, where the bits have the same meaning as shown in Figure 5-8. For example, the sequence [start, bit 0, bit 1, bit 2]=[0,1,0,0] grants DMA channel 1 to the requesting device, and the sequence [start, bit 0, bit 1, bit 2]=[0,0,1,1] grants DMA channel 6 to the requesting device.

All PCI DMA expansion agents must use the channel passing protocol described above. They must also work as follows:

1. If a PCI DMA expansion agent has more than one request active, it must resend the request serial protocol after one of the requests has been granted the bus and it has completed its transfer. The expansion device should drive its REQ# inactive for two clocks and then transmit the serial channel passing protocol again, even if there are no new requests from the PCI expansion agent to ICH. For example: If a PCI expansion agent had active requests for DMA Channel 1 and Channel 5, it would pass this information to the ICH through the expansion channel passing protocol. If after receiving GNT# (assume for CH5) and having the device finish its transfer (device stops driving request to PCI expansion agent) it would then need to re-transmit the expansion channel passing protocol to inform ICH that DMA channel 1 was still requesting the bus, even if that was the only request the expansion device had pending.
2. If a PCI DMA expansion agent has a request go inactive before ICH asserts GNT#, it must resend the expansion channel passing protocol to update ICH with this new request information. For example: If a PCI expansion agent has DMA channel 1 and 2 requests pending, it sends them serially to ICH using the expansion channel passing protocol. If, however, DMA channel 1 goes inactive into the expansion agent before the expansion agent receives a GNT# from ICH, the expansion agent **MUST** pull its REQ# line high for ONE clock and resend the expansion channel passing information with only DMA channel 2 active. Note that ICH does not do anything special to catch this case because a DREQ going inactive before a DACK# is received is not allowed in the ISA DMA protocol and, therefore, does not need to work properly in this protocol either. This requirement is needed to be able to support Plug-n-Play ISA devices that toggle DREQ# lines to determine if those lines are free in the system.
3. If a PCI expansion agent has sent its serial request information and receives a new DMA request before receiving GNT# the agent must resend the serial request with the new request active. For example, if a PCI expansion agent has already passed requests for DMA channel 1 and 2 and sees DREQ 3 active before a GNT# is received, the device must pull its REQ# line high for one clock and resend the expansion channel passing information with all three channels active.

The three cases above require the following functionality in the PCI DMA expansion device:

1. Drive REQ# inactive for one clock to signal new request information.
2. Drive REQ# inactive for two clocks to signal that a request that had been granted the bus has gone inactive.
3. The REQ# and GNT# state machines must run independently and concurrently (i.e., a GNT# could be received while in the middle of sending a serial REQ# or a GNT# could be active while REQ# is inactive).

## 5.4.2 PCI DMA Expansion Cycles

The ICH's support of the PC/PCI DMA Protocol currently consists of four types of cycles: Memory to I/O, I/O to Memory, Verify, and ISA Master cycles. ISA Masters are supported through the use of a DMA channel that has been programmed for cascade mode.

The DMA controller does a two cycle transfer (a load followed by a store) as opposed to the ISA "fly-by" cycle for PC/PCI DMA agents. The memory portion of the cycle generates a PCI memory read or memory write bus cycle, its address representing the selected memory.

The I/O portion of the DMA cycle generates a PCI I/O cycle to one of four I/O addresses (Table 5-9). Note that these cycles must be qualified by an active GNT# signal to the requesting device.

Table 5-9. DMA Cycle vs. I/O Address

| DMA Cycle Type | DMA I/O Address | PCI Cycle Type |
|----------------|-----------------|----------------|
| Normal         | 00h             | I/O Read/Write |
| Normal TC      | 04h             | I/O Read/Write |
| Verify         | 0C0h            | I/O Read       |
| Verify TC      | 0C4h            | I/O Read       |

### 5.4.3 DMA Addresses

The memory portion of the cycle generates a PCI memory read or memory write bus cycle, its address representing the selected memory. The I/O portion of the DMA cycle generates a PCI I/O cycle to one of the four I/O addresses listed in Table 5-9.

### 5.4.4 DMA Data Generation

The data generated by PC/PCI devices on I/O reads when they have an active GNT# is on the lower two bytes of the PCI AD bus. Table 5-10 lists the PCI pins that the data appears on for 8 and 16 bit channels. Each I/O read results in one memory write, and each memory read results in one I/O write. If the I/O device is 8 bit, the ICH performs an 8 bit memory write. The ICH does not assemble the I/O read into a dWord for writing to memory. Similarly, the ICH will not disassemble a DWord read from memory to the I/O device.

Table 5-10. PCI Data Bus vs. DMA I/O port size

| PCI DMA I/O Port Size | PCI Data Bus Connection |
|-----------------------|-------------------------|
| Byte                  | AD[7:0]                 |
| Word                  | AD[15:0]                |

### 5.4.5 DMA Byte Enable Generation

The byte enables generated by the ICH on I/O reads and writes must correspond to the size of the I/O device. Table 5-11 defines the byte enables asserted for 8 and 16 bit DMA cycles.

Table 5-11. DMA I/O Cycle Width vs. BE[3:0]#

| BE[3:0]# | Description                        |
|----------|------------------------------------|
| 1110b    | 8-bit DMA I/O Cycle: Channels 0-3  |
| 1100b    | 16-bit DMA I/O Cycle: Channels 5-7 |

**NOTE:** For verify cycles the value of the Byte Enables (BEs) is a "don't care".

## 5.4.6 DMA Cycle Termination

DMA cycles are terminated when a terminal count is reached in the DMA controller and the channel is not in autoinitialize mode, or when the PC/PCI device deasserts its request. The PC/PCI device must follow explicit rules when deasserting its request, or the ICH may not see it in time and run an extra I/O and memory cycle.

The PC/PCI device must deassert its request 7 PCICLKs before it generates TRDY# on the I/O read or write cycle, or the ICH is allowed to generate another DMA cycle. For transfers to memory, this means that the memory portion of the cycle will be run without an asserted PC/PCI REQ#.

## 5.5 LPC DMA

DMA on LPC is handled through the use of the LDRQ# lines from peripherals and special encodings on LAD[3:0] from the host. Single, Demand, Verify, and Increment modes are supported on the LPC interface. Channel's 0 - 3 are 8 bit channels. Channel's 5 - 7 are 16 bit channels. Channel 4 is reserved as a generic bus master request.

### 5.5.1 Asserting DMA Requests

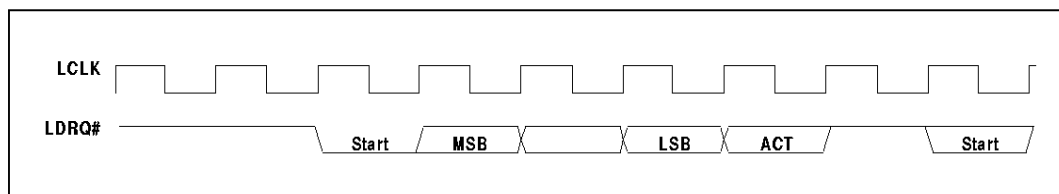
Peripherals that need DMA service encode their requested channel number on the LDRQ# signal. To simplify the protocol, each peripheral on the LPC Interface has its own dedicated LDRQ# signal (they may not be shared between two separate peripherals). The ICH has two LDRQ# inputs, allowing at least two devices to support DMA or bus mastering.

LDRQ# is synchronous with LCLK (PCI clock). As shown in Figure 5-9 the peripheral uses the following serial encoding sequence:

- Peripheral starts the sequence by asserting LDRQ# low (start bit). LDRQ# is high during idle conditions.
- The next 3 bits contain the encoded DMA channel number (MSB first).
- The next bit (ACT) indicates whether the request for the indicated DMA channel is active or inactive. The ACT bit is a 1 (high) to indicate if it is active and 0 (low) if it is inactive. The case where ACT is low is rare, and is only used to indicate that a previous request for that channel is being abandoned.
- After the active/inactive indication, the LDRQ# signal must go high for at least 1 clock. After that one clock, LDRQ# signal can be brought low to the next encoding sequence.

If another DMA channel also needs to request a transfer, another sequence can be sent on LDRQ#. For example, if an encoded request is sent for channel 2, and then channel 3 needs a transfer before the cycle for channel 2 is run on the interface, the peripheral can send the encoded request for channel 3. This allows multiple DMA agents behind an I/O device to request use of the LPC interface, and the I/O device does not need to self-arbitrate before sending the message.

Figure 5-9. DMA Request Assertion Through LDRQ#



## 5.5.2 Abandoning DMA Requests

DMA Requests can be de-asserted in two fashions: on error conditions by sending an LDRQ# message with the 'ACT' bit set to '0', or normally through a SYNC field during the DMA transfer. This section describes boundary conditions where the DMA request needs to be removed prior to a data transfer.

There may be some special cases where the peripheral desires to abandon a DMA transfer. The most likely case of this occurring is due to a floppy disk controller which has overrun or underrun its FIFO, or software stopping a device prematurely.

In these cases, the peripheral wishes to stop further DMA activity. It may do so by sending an LDRQ# message with the ACT bit as '0'. However, since the DMA request was seen by the ICH, there is no guarantee that the cycle has not been granted and will shortly run on LPC. Therefore, peripherals must take into account that a DMA cycle may still occur. The peripheral can choose not to respond to this cycle, in which case the host will abort it, or it can choose to complete the cycle normally with any random data.

This method of DMA deassertion should be prevented whenever possible, to limit boundary conditions both on the ICH and the peripheral.

## 5.5.3 General Flow of DMA Transfers

Arbitration for DMA channels is performed through the 8237 within the host. Once the host has won arbitration on behalf of a DMA channel assigned to LPC, it asserts LFRAME# on the LPC Interface and begins the DMA transfer. The general flow for a basic DMA transfer is as follows:

1. ICH starts transfer by asserting '0000b' on LAD[3:0] with LFRAME# asserted.
2. ICH asserts 'cycle type' of DMA, direction based on DMA transfer direction.
3. ICH asserts channel number and, if applicable, terminal count.
4. ICH indicates the size of the transfer: 8 or 16 bits.
5. If a DMA read...
  - The ICH drives the first 8 bits of data and turns the bus around.
  - The peripheral acknowledges the data with a valid SYNC.
  - If a 16 bit transfer, the process is repeated for the next 8 bits.
6. If a DMA write...
  - The ICH turns the bus around and waits for data.
  - The peripheral indicates data ready through SYNC and transfers the first byte.
  - If a 16 bit transfer, the peripheral indicates data ready and transfers the next byte.
7. The peripheral turns around the bus.



### 5.5.4 Terminal Count

Terminal count is communicated through LAD[3] on the same clock that DMA channel is communicated on LAD[2:0]. This field is the CHANNEL field. Terminal count indicates the last byte of transfer, based upon the size of the transfer.

For example, on an 8 bit transfer size (SIZE field is '00b'), if the TC bit is set, then this is the last byte. On a 16 bit transfer (SIZE field is '01b'), if the TC bit is set, then the second byte is the last byte. The peripheral, therefore, must internalize the TC bit when the CHANNEL field is communicated, and only signal TC when the last byte of that transfer size has been transferred.

### 5.5.5 Verify Mode

Verify mode is supported on the LPC interface. A verify transfer to the peripheral is similar to a DMA write, where the peripheral is transferring data to main memory. The indication from the host is the same as a DMA write, so the peripheral will be driving data onto the LPC interface. However, the host does not transfer this data into main memory.

### 5.5.6 DMA Request De-Assertion

An end of transfer is communicated to the ICH through a special SYNC field transmitted by the peripheral. An LPC device must not attempt to signal the end of a transfer by de-asserting LDREQ#. If a DMA transfer is several bytes, such as a transfer from a demand mode device, the ICH needs to know when to de-assert the DMA request based on the data currently being transferred.

The DMA agent uses a SYNC encoding on each byte of data being transferred, which indicates to the ICH whether this is the last byte of transfer or if more bytes are requested. To indicate the last byte of transfer, the peripheral uses a SYNC value of '0000b' (ready with no error), or '1010b' (ready with error). These encodings tell the ICH that this is the last piece of data transferred on a DMA read (ICH to peripheral), or the byte which follows is the last piece of data transferred on a DMA write (peripheral to ICH).

When the ICH sees one of these two encodings, it ends the DMA transfer after this byte and de-asserts the DMA request to the 8237. Therefore, if the ICH indicated a 16 bit transfer, the peripheral can end the transfer after one byte by indicating a SYNC value of '0000b' or '1010b'. The ICH does not attempt to transfer the second byte, and de-asserts the DMA request internally.

If the peripheral indicates a '0000b' or '1010b' SYNC pattern on the last byte of the indicated size, then the ICH only de-asserts the DMA request to the 8237 since it does not need to end the transfer.

If the peripheral wishes to keep the DMA request active, then it uses a SYNC value of '1001b' (ready plus more data). This tells the 8237 that more data bytes are requested after the current byte has been transferred, so the ICH keeps the DMA request active to the 8237. Therefore, on an 8 bit transfer size, if the peripheral indicates a SYNC value of '1001b' to the ICH, the data is transferred and the DMA request remains active to the 8237. At a later time, the ICH then comes back with another START - CYCTYPE - CHANNEL - SIZE etc. combination to initiate another transfer to the peripheral.

The peripheral must not assume that the next START indication from the ICH is another grant to the peripheral if it had indicated a SYNC value of '1001b'. On a single mode DMA device, the 8237 re-arbitrates after every transfer. Only demand mode DMA devices can be guaranteed that they receive the next START indication from the ICH.

**Note:** Indicating a '0000b' or '1010b' encoding on the SYNC field of an odd byte of a 16 bit channel (first byte of a 16 bit transfer) is an error condition.

**Note:** The host stops the transfer on the LPC bus as indicated, fills the upper byte with random data on DMA writes (peripheral to memory), and indicates to the 8237 that the DMA transfer occurred, incrementing the 8237's address and decrementing its byte count.

### 5.5.7 SYNC Field / LDRQ# Rules

Since DMA transfers on LPC are requested through an LDRQ# assertion message, and are ended through a SYNC field during the DMA transfer, the peripheral must obey the following rule when initiating back-to-back transfers from a DMA channel.

The peripheral must not assert another message for 8 LCLKs after a de-assertion is indicated through the SYNC field. This is needed to allow the 8237, which typically runs off a much slower internal clock, to see a message de-asserted before it is re-asserted so that it can arbitrate to the next agent.

Under default operation, the host only performs 8 bit transfers on 8 bit channels and 16 bit transfers on 16 bit channels.

The method by which this communication between host and peripheral through system BIOS is performed is beyond the scope of this specification. Since the LPC host and LPC peripheral are motherboard devices, no "plug-n-play" registry is required.

The peripheral must not assume that the host is able to perform transfer sizes that are larger than the size allowed for the DMA channel, and is willing to accept a SIZE field that is smaller than what it may currently have buffered.

To that end, it is recommended that future devices which may appear on the LPC bus, which require higher bandwidth than 8 bit or 16 bit DMA allow, do so with a bus mastering interface and not rely on the 8237.

## 5.6 8254 Timers (D31:F0)

The ICH contains three counters which have fixed uses. All registers and functions associated with the 8254 timers are in the Core well. The 8254 unit is clocked by a 14.31818 MHz clock.

### Counter 0, System Timer

This counter functions as the system timer by controlling the state of **IRQ0** and is typically programmed for Mode 3 operation. The counter produces a square wave with a period equal to the product of the counter period (838 ns) and the initial count value. The counter loads the initial count value one counter period after software writes the count value to the counter I/O address. The counter initially asserts **IRQ0** and decrements the count value by two each counter period. The counter negates **IRQ0** when the count value reaches 0. It then reloads the initial count value and again decrements the initial count value by two each counter period. The counter then asserts **IRQ0** when the count value reaches 0, reloads the initial count value, and repeats the cycle, alternately asserting and negating **IRQ0**.

### Counter 1, Refresh Request Signal

This counter provides the refresh request signal and is typically programmed for Mode 2 operation. The counter negates refresh request for one counter period (838 ns) during each count cycle. The initial count value is loaded one counter period after being written to the counter I/O address. The counter initially asserts refresh request, and negates it for 1 counter period when the count value reaches 1. The counter then asserts refresh request and continues counting from the initial count value.

### Counter 2, Speaker Tone

This counter provides the speaker tone and is typically programmed for Mode 3 operation. The counter provides a speaker frequency equal to the counter clock frequency (1.193 MHz) divided by the initial count value. The speaker must be enabled by a write to port 061h (see **NMI Status and Control ports**).

### 5.6.1 Timer Programming

The counter/timers are programmed in the following fashion:

1. Write a control word to select a counter
2. Write an initial count for that counter.
3. Load the least and/or most significant bytes (as required by Control Word bits 5, 4) of the 16-bit counter.
4. Repeat with other counters

Only two conventions need to be observed when programming the counters. First, for each counter, the control word must be written before the initial count is written. Second, the initial count must follow the count format specified in the control word (least significant byte only, most significant byte only, or least significant byte and then most significant byte).

A new initial count may be written to a counter at any time without affecting the counter's programmed mode. Counting is affected as described in the mode definitions. The new count must follow the programmed count format.

If a counter is programmed to read/write two-byte counts, the following precaution applies: A program must not transfer control between writing the first and second byte to another routine which also writes into that same counter. Otherwise, the counter is loaded with an incorrect count.

The Control Word Register at port 43h controls the operation of all three counters. Several commands are available:

- **Control Word Command.** Specifies which counter to read or write, the operating mode, and the count format (binary or BCD).
- **Counter Latch Command.** Latches the current count so that it can be read by the system. The countdown process continues.
- **Read Back Command.** Reads the count value, programmed mode, the current state of the OUT pins, and the state of the Null Count Flag of the selected counter.

Table 5-12 lists the six operating modes for the interval counters.

**Table 5-12. Counter Operating Modes**

| Mode | Function                             | Description   |
|------|--------------------------------------|---|
| 0    | Out signal on end of count (=0)      | Output is '0'. When count goes to 0, output goes to '1' and stays at '1' until counter is reprogrammed.   |
| 1    | Hardware retriggerable one-shot      | Output is '0'. When count goes to 0, output goes to '1' for one clock time.   |
| 2    | Rate generator (divide by n counter) | Output is '1'. Output goes to '0' for one clock time, then back to '1' and counter is reloaded.   |
| 3    | Square wave output                   | Output is '1'. Output goes to '0' when counter rolls over, and counter is reloaded. Output goes to '1' when counter rolls over, and counter is reloaded, etc. |
| 4    | Software triggered strobe            | Output is '1'. Output goes to '0' when count expires for one clock time.  |
| 5    | Hardware triggered strobe            | Output is '1'. Output goes to '0' when count expires for one clock time.  |

## 5.6.2 Reading from the Interval Timer

It is often desirable to read the value of a counter without disturbing the count in progress. There are three methods for reading the counters: a simple read operation, counter Latch Command, and the Read-Back Command. Each is explained below.

With the simple read and counter latch command methods, the count must be read according to the programmed format; specifically, if the counter is programmed for two byte counts, two bytes must be read. The two bytes do not have to be read one right after the other. Read, write, or programming operations for other counters may be inserted between them.

### Simple Read

The first method is to perform a simple read operation. The counter is selected through port 40h (counter 0), 41h (counter 1), or 42h (counter 2).

**Note:** Performing a direct read from the counter does not return a determinate value, because the counting process is asynchronous to read operations. However, in the case of counter 2, the count can be stopped by writing to the GATE bit in port 61h.

### Counter Latch Command

The Counter Latch Command, written to port 43h, latches the count of a specific counter at the time the command is received. This command is used to ensure that the count read from the counter is accurate, particularly when reading a two-byte count. The count value is then read from each counter's Count Register as was programmed by the Control Register.

The count is held in the latch until it is read or the counter is reprogrammed. The count is then unlatched. This allows reading the contents of the counters on the fly without affecting counting in progress. Multiple Counter Latch Commands may be used to latch more than one counter. Counter Latch Commands do not affect the programmed mode of the counter in any way.

If a Counter is latched and then, some time later, latched again before the count is read, the second Counter Latch Command is ignored. The count read is the count at the time the first Counter Latch Command was issued.

### Read Back Command

The Read Back Command, written to port 43h, latches the count value, programmed mode, and current states of the OUT pin and Null Count flag of the selected counter or counters. The value of the counter and its status may then be read by I/O access to the counter address.

The Read Back Command may be used to latch multiple counter outputs at one time. This single command is functionally equivalent to several counter latch commands, one for each counter latched. Each counter's latched count is held until it is read or reprogrammed. Once read, a counter is unlatched. The other counters remain latched until they are read. If multiple count Read Back Commands are issued to the same counter without reading the count, all but the first are ignored.

The Read Back Command may additionally be used to latch status information of selected counters. The status of a counter is accessed by a read from that counter's I/O port address. If multiple counter status latch operations are performed without reading the status, all but the first are ignored.

Both count and status of the selected counters may be latched simultaneously. This is functionally the same as issuing two consecutive, separate Read Back Commands. If multiple count and/or status Read Back Commands are issued to the same counters without any intervening reads, all but the first are ignored.

If both count and status of a counter are latched, the first read operation from that counter returns the latched status, regardless of which was latched first. The next one or two reads, depending on whether the counter is programmed for one or two type counts, return the latched count. Subsequent reads return unlatched count.

## 5.7 8259 Interrupt Controllers (PIC) (D31:F0)

The ICH incorporates the functionality of two 8259 interrupt controllers that provide system interrupts for the ISA compatible interrupts. These interrupts are: system timer, keyboard controller, serial ports, parallel ports, floppy disk, IDE, mouse, and DMA channels. In addition, this interrupt controller can support the PCI based interrupts, by mapping the PCI interrupt onto the compatible ISA interrupt line. Each 8259 core supports 8 interrupts, numbered 0 - 7. Table 5-13 shows how the cores are connected

**Table 5-13. Interrupt Controller Core Connections**

| 8259   | 8259 input | Typical Interrupt Source | Connected Pin / Function                                 |
|--------|------------|--------------------------|--|
| Master | 0          | Internal                 | Internal Timer / Counter 0 output                        |
|        | 1          | Keyboard                 | IRQ1 via SERIRQ  |
|        | 2          | Internal                 | Slave Controller INTR output                             |
|        | 3          | Serial Port A            | IRQ3 via SERIRQ  |
|        | 4          | Serial Port B            | IRQ4 via SERIRQ  |
|        | 5          | Parallel Port / Generic  | IRQ5 via SERIRQ  |
|        | 6          | Floppy Disk              | IRQ6 via SERIRQ  |
|        | 7          | Parallel Port / Generic  | IRQ7 via SERIRQ  |
| Slave  | 0          | Internal Real Time Clock | Internal RTC   |
|        | 1          | Generic                  | IRQ9 via SERIRQ  |
|        | 2          | Generic                  | IRQ10 via SERIRQ   |
|        | 3          | Generic                  | IRQ11 via SERIRQ   |
|        | 4          | PS/2 Mouse               | IRQ12 via SERIRQ   |
|        | 5          | Internal                 | State Machine output based on processor FERR# assertion. |
|        | 6          | Primary IDE cable        | IRQ14 from input signal or via SERIRQ                    |
|        | 7          | Secondary IDE Cable      | IRQ15 from input signal or via SERIRQ                    |

The ICH cascades the slave controller onto the master controller through master controller interrupt input 2. This means there are only 15 possible interrupts for the ICH PIC. Interrupts can individually be programmed to be edge or level, except for IRQ0, IRQ2, IRQ8#, and IRQ13.

**Note:** Previous devices internally latched IRQ12 and IRQ1 and required a port 60h read to clear the latch. The ICH can be programmed to latch IRQ12 or IRQ1 (see bit 11 and bit 12 in General Control Register, Section 8.1.21).

### 5.7.1 Interrupt Handling

#### Generating Interrupts

The PIC interrupt sequence involves three bits, from the IRR, ISR, and IMR, for each interrupt level. These bits are used to determine the interrupt vector returned, and status of any other pending interrupts. Table 5-14 defines the IRR, ISR and IMR.

**Table 5-14. Interrupt Status Registers**

| Bit | Description   |
|-----|---|
| IRR | <b>Interrupt Request Register.</b> This bit is set on a low to high transition of the interrupt line in edge mode, and by an active high level in level mode. This bit is set whether or not the interrupt is masked. However, a masked interrupt does not generate INTR. |
| ISR | <b>Interrupt Service Register.</b> This bit is set, and the corresponding IRR bit cleared, when an interrupt acknowledge cycle is seen, and the vector returned is for that interrupt.  |
| IMR | <b>Interrupt Mask Register.</b> This bit determines whether an interrupt is masked. Masked interrupts do not generate INTR.   |

### Acknowledging Interrupts

The processor generates an interrupt acknowledge cycle which is translated by the host bridge into a PCI Interrupt Acknowledge Cycle to the ICH. The PIC translates this command into two internal INTA# pulses expected by the 8259 cores. The PIC uses the first internal INTA# pulse to freeze the state of the interrupts for priority resolution. On the second INTA# pulse, the master or slave sends the interrupt vector to the processor with the acknowledged interrupt code. This code is based upon bits [7:3] of the corresponding ICW2 register, combined with three bits representing the interrupt within that controller.

**Table 5-15. Content of Interrupt Vector Byte**

| Master,Slave Interrupt | Bits [7:3] | Bits [2:0] |
|------------------------|------------|------------|
| IRQ7,15                | ICW2[7:3]  | 111        |
| IRQ6,14                |            | 110        |
| IRQ5,13                |            | 101        |
| IRQ4,12                |            | 100        |
| IRQ3,11                |            | 011        |
| IRQ2,10                |            | 010        |
| IRQ1,9                 |            | 001        |
| IRQ0,8                 |            | 000        |

### Hardware/Software Interrupt Sequence

1. One or more of the Interrupt Request lines (IRQ) are raised high in edge mode, or seen high in level mode, setting the corresponding IRR bit.
2. The PIC sends INTR active (low) to the processor if an asserted interrupt is not masked.
3. The processor acknowledges the INTR and responds with an interrupt acknowledge cycle. The cycle is translated into a PCI interrupt acknowledge cycle by the host bridge. This command is broadcast over PCI by the ICH.
4. Upon observing its own interrupt acknowledge cycle on PCI, the ICH converts it into the two cycles that the internal 8259 pair can respond to. Each cycle appears as an interrupt acknowledge pulse on the internal INTA# pin of the cascaded interrupt controllers.
5. Upon receiving the first internally generated INTA# pulse, the highest priority ISR bit is set and the corresponding IRR bit is reset. On the trailing edge of the first pulse, a slave identification code is broadcast by the master to the slave on a private, internal three bit wide bus. The slave controller uses these bits to determine if it must respond with an interrupt vector during the second INTA# pulse.
6. Upon receiving the second internally generated INTA# pulse, the PIC returns the interrupt vector. If no interrupt request is present because the request was too short in duration, the PIC returns vector 7 from the master controller.
7. This completes the interrupt cycle. In AEOI mode the ISR bit is reset at the end of the second INTA# pulse. Otherwise, the ISR bit remains set until an appropriate EOI command is issued at the end of the interrupt subroutine.

## 5.7.2 Initialization Command Words (ICWx)

Before operation can begin, each 8259 must be initialized. In the ICH, this is a four byte sequence. The four initialization command words are referred to by their acronyms: ICW1, ICW2, ICW3, and ICW4.

The base address for each 8259 initialization command word is a fixed location in the I/O memory space: 20h for the master controller, and A0h for the slave controller.

### ICW1

An I/O write to the master or slave controller base address with data bit 4 equal to 1 is interpreted as a write to ICW1. Upon sensing this write, the ICH PIC expects three more byte writes to 21h for the master controller, or A1h for the slave controller, to complete the ICW sequence.

A write to ICW1 starts the initialization sequence during which the following automatically occur:

1. Following initialization, an interrupt request (IRQ) input must make a low-to-high transition to generate an interrupt.
2. The Interrupt Mask Register is cleared.
3. IRQ7 input is assigned priority 7.
4. The slave mode address is set to 7.
5. Special Mask Mode is cleared and Status Read is set to IRR.



## ICW2

The second write in the sequence, ICW2, is programmed to provide bits [7:3] of the interrupt vector that is released during an interrupt acknowledge. A different base is selected for each interrupt controller.

## ICW3

The third write in the sequence, ICW3, has a different meaning for each controller.

- For the master controller, ICW3 is used to indicate which IRQ input line is used to cascade the slave controller. Within the ICH, IRQ2 is used. Therefore, bit 2 of ICW3 on the master controller is set to a 1, and the other bits are set to 0's.
- For the slave controller, ICW3 is the slave identification code used during an interrupt acknowledge cycle. On interrupt acknowledge cycles, the master controller broadcasts a code to the slave controller if the cascaded interrupt won arbitration on the master controller. The slave controller compares this identification code to the value stored in its ICW3, and if it matches, the slave controller assumes responsibility for broadcasting the interrupt vector.

## ICW4

The final write in the sequence, ICW4, must be programmed both controllers. At the very least, bit 0 must be set to a 1 to indicate that the controllers are operating in an Intel Architecture-based system.

## 5.7.3 Operation Command Words (OCW)

These command words reprogram the Interrupt Controller to operate in various interrupt modes.

- OCW1 masks and unmasks interrupt lines.
- OCW2 controls the rotation of interrupt priorities when in rotating priority mode, and controls the EOI function.
- OCW3 is sets up ISR/IRR reads, enables/disables the Special Mask Mode SMM, and enables/disables polled interrupt mode.

## 5.7.4 Modes of Operation

### Fully Nested Mode

In this mode, interrupt requests are ordered in priority from 0 through 7, with 0 being the highest. When an interrupt is acknowledged, the highest priority request is determined and its vector placed on the bus. Additionally, the ISR for the interrupt is set. This ISR bit remains set until: the processor issues an EOI command immediately before returning from the service routine; or if in AEOI mode, on the trailing edge of the second INTA#. While the ISR bit is set, all further interrupts of the same or lower priority are inhibited, while higher levels generate another interrupt.

Interrupt priorities can be changed in the rotating priority mode.

### Special Fully Nested Mode

This mode is used in the case of a system where cascading is used, and the priority has to be conserved within each slave. In this case, the special fully nested mode is programmed to the master controller. This mode is similar to the fully nested mode with the following exceptions:

- When an interrupt request from a certain slave is in service, this slave is not locked out from the master's priority logic and further interrupt requests from higher priority interrupts within the slave is recognized by the master and initiates interrupts to the processor. In the normal nested mode, a slave is masked out when its request is in service.
- When exiting the Interrupt Service routine, software has to check whether the interrupt serviced was the only one from that slave. This is done by sending a Non-Specific EOI command to the slave and then reading its ISR. If it is 0, a non-specific EOI can also be sent to the master.

### Automatic Rotation Mode (Equal Priority Devices)

In some applications, there are a number of interrupting devices of equal priority. Automatic rotation mode provides for a sequential 8-way rotation. In this mode, a device receives the lowest priority after being serviced. In the worst case, a device requesting an interrupt has to wait until each of seven other devices are serviced at most once.

There are two ways to accomplish automatic rotation using OCW2; the Rotation on Non-Specific EOI Command (R=1, SL=0, EOI=1) and the Rotate in Automatic EOI Mode which is set by (R=1, SL=0, EOI=0).

### Specific Rotation Mode (Specific Priority)

Software can change interrupt priorities by programming the bottom priority. For example, if IRQ5 is programmed as the bottom priority device, then IRQ6 is the highest priority device. The Set Priority Command is issued in OCW2 to accomplish this, where: R=1, SL=1, and LO-L2 is the binary priority level code of the bottom priority device.

In this mode, internal status is updated by software control during OCW2. However, it is independent of the EOI command. Priority changes can be executed during an EOI command by using the Rotate on Specific EOI Command in OCW2 (R=1, SL=1, EOI=1 and LO-L2=IRQ level to receive bottom priority).

### Poll Mode

Poll Mode can be used to conserve space in the interrupt vector table. Multiple interrupts that can be serviced by one interrupt service routine do not need separate vectors if the service routine uses the poll command. Polled Mode can also be used to expand the number of interrupts. The polling interrupt service routine can call the appropriate service routine, instead of providing the interrupt vectors in the vector table. In this mode, the INTR output is not used and the microprocessor internal Interrupt Enable flip-flop is reset, disabling its interrupt input. Service to devices is achieved by software using a Poll Command.

The Poll command is issued by setting P=1 in OCW3. The PIC treats its next I/O read as an interrupt acknowledge, sets the appropriate ISR bit if there is a request, and reads the priority level. Interrupts are frozen from the OCW3 write to the I/O read. The byte returned during the I/O read contains a '1' in bit 7 if there is an interrupt, and the binary code of the highest priority level in bits 2:0.

### Cascade Mode

The PIC in the ICH has one master 8259 and one slave 8259 cascaded onto the master through IRQ2. This configuration can handle up to 15 separate priority levels. The master controls the slaves through a three bit internal bus. In the ICH, when the master drives 010b on this bus, the slave controller takes responsibility for returning the interrupt vector. An EOI Command must be issued twice: once for the master and once for the slave.

### Edge and Level Triggered Mode

In ISA systems this mode is programmed using bit 3 in ICW1, which sets level or edge for the entire controller. In the ICH, this bit is disabled and a new register for edge and level triggered mode selection, per interrupt input, is included. This is the Edge/Level control Registers ELCR1 and ELCR2.

If an ELCR bit is '0', an interrupt request is recognized by a low to high transition on the corresponding IRQ input. The IRQ input can remain high without generating another interrupt. If an ELCR bit is '1', an interrupt request is recognized by a high level on the corresponding IRQ input and there is no need for an edge detection. The interrupt request must be removed before the EOI command is issued to prevent a second interrupt from occurring.

In both the edge and level triggered modes, the IRQ inputs must remain active until after the falling edge of the first internal INTA#. If the IRQ input goes inactive before this time, a default IRQ7 vector is returned.

### End of Interrupt Operations

An EOI can occur in one of two fashions: by a command word write issued to the PIC before returning from a service routine, the EOI command; or automatically when AEOI bit in ICW1 is set to 1.

### Normal End of Interrupt

In Normal EOI, software writes an EOI command before leaving the interrupt service routine to mark the interrupt as completed. There are two forms of EOI commands: Specific and Non-Specific. When a Non-Specific EOI command is issued, the PIC clears the highest ISR bit of those that are set to 1. Non-Specific EOI is the normal mode of operation of the PIC within the ICH, as the interrupt being serviced currently is the interrupt entered with the interrupt acknowledge. When the PIC is operated in modes which preserve the fully nested structure, software can determine which ISR bit to clear by issuing a Specific EOI. An ISR bit that is masked is not be cleared by a Non-Specific EOI if the PIC is in the Special Mask Mode. An EOI command must be issued for both the master and slave controller.

### Automatic End of Interrupt Mode

In this mode, the PIC automatically performs a Non-Specific EOI operation at the trailing edge of the last interrupt acknowledge pulse. From a system standpoint, this mode should be used only when a nested multi-level interrupt structure is not required within a single PIC. The AEOI mode can only be used in the master controller and not the slave controller.

## 5.7.5 Masking Interrupts

### Masking on an Individual Interrupt Request

Each interrupt request can be masked individually by the Interrupt Mask Register (IMR). This register is programmed through OCW1. Each bit in the IMR masks one interrupt channel. Masking IRQ2 on the master controller masks all requests for service from the slave controller.

### Special Mask Mode

Some applications may require an interrupt service routine to dynamically alter the system priority structure during its execution under software control. For example, the routine may wish to inhibit lower priority requests for a portion of its execution but enable some of them for another portion.

The Special Mask Mode enables all interrupts not masked by a bit set in the Mask Register. Normally, when an interrupt service routine acknowledges an interrupt without issuing an EOI to clear the ISR bit, the interrupt controller inhibits all lower priority requests. In the Special Mask Mode, any interrupts may be selectively enabled by loading the Mask Register with the appropriate pattern. The special Mask Mode is set by OCW3 where: SSMM=1, SMM=1, and cleared where SSMM=1, SMM=0.

## 5.7.6 Steering PCI Interrupts

The ICH can be programmed to allow PIRQA#-PIRQD# to be internally routed to interrupts 3-7, 9-12, 14 or 15. The assignment is programmable through the PIRQx Route Control registers, located at 60-63h in function 0. One or more PIRQx# lines can be routed to the same IRQx input. If interrupt steering is not required, the Route Registers can be programmed to disable steering.

The PIRQx# lines are defined as active low, level sensitive to allow multiple interrupts on a PCI Board to share a single line across the connector. When a PIRQx# is routed to specified IRQ line, software must change the IRQ's corresponding ELCR bit to level sensitive mode. The ICH internally inverts the PIRQx# line to send an active high level to the PIC. When a PCI interrupt is routed onto the PIC, the selected IRQ can no longer be used by an ISA device (through SERIRQ). However, active low non-ISA interrupts can share their interrupt with PCI interrupts.

## 5.8 Advanced Interrupt Controller (APIC) (D31:F0)

In addition to the standard ISA compatible interrupt controller (PIC) described in the previous chapter, the ICH incorporates the Advanced Programmable Interrupt Controller (APIC). While the standard interrupt controller is intended for use in a uni-processor system, APIC can be used in either a uni-processor or multi-processor system.

### 5.8.1 Interrupt Handling

The I/O APIC handles interrupts very differently than the 8259. Briefly, these differences are:

- **Method of Interrupt Transmission.** The I/O APIC transmits interrupts through a three wire bus, and interrupts are handled without the need for the processor to run an interrupt acknowledge cycle.
- **Interrupt Priority.** The priority of interrupts in the I/O APIC is independent of the interrupt number. For example, interrupt 10 can be given a higher priority than interrupt 3.
- **More Interrupts.** The I/O APIC in the ICH supports a total of 24 interrupts.
- **Multiple Interrupt Controllers.** The I/O APIC interrupt transmission protocol has an arbitration phase, which allows for multiple I/O APICs in the system with their own interrupt vectors. The ICH I/O APIC must arbitrate for the APIC bus before transmitting its interrupt message.

### 5.8.2 Interrupt Mapping

The I/O APIC within the ICH supports 24 APIC interrupts. Each interrupt has its own unique vector assigned by software. The interrupt vectors are mapped as shown in Table 5-16.

Table 5-16. APIC interrupt Mapping

| I/O APIC Input    | Interrupt                | Comment  |
|-------------------|--------------------------|--|
| Interrupt 0       | 8259 INTR                |  |
| Interrupt 1       | ISA Interrupt 1          |  |
| Interrupt 2       | ISA Interrupt 0          | Counter 0 output of 8254                           |
| Interrupts 3 - 12 | ISA IRQ3-7,IRQ8#,IRQ9-12 | IRQ8# is inverted internally                       |
| Interrupt 13      | ISA interrupt 13         | Internally generated off of FERR#                  |
| Interrupt 14 - 15 | ISA Interrupt 14-15      | Comes from IRQ[14:15] input signals or from SERIRQ |
| Interrupt 16      | PCI Interrupt A          | Comes from PIRQA input signal or SERIRQ            |
| Interrupt 17      | PCI Interrupt B          | Comes from PIRQB input signal or SERIRQ            |
| Interrupt 18      | PCI Interrupt C          | Comes from PIRQC input signal or SERIRQ            |
| Interrupt 19      | PCI Interrupt D          | Comes from PIRQD input signal or SERIRQ            |
| Interrupt 20      | IRQ20                    | Can come from internal modules                     |
| Interrupt 21      | IRQ21                    | Can come from internal modules                     |
| Interrupt 22      | IRQ22                    | Can come from internal modules                     |
| Interrupt 23      | IRQ23                    | Can come from internal modules                     |

## 5.8.3 APIC Bus Functional Description

### 5.8.3.1 Physical Characteristics of APIC

The APIC bus is a 3-wire synchronous bus connecting all I/O and local APICs. Two of these wires are used for data transmission, and one wire is a clock. For bus arbitration, the APIC uses only one of the data wires. The bus is logically a wire-OR and electrically an open-drain connection providing for both bus use arbitration and arbitration for lowest priority. The APIC bus speed can run from 16.67 MHz to 33 MHz.

### 5.8.3.2 APIC Bus Arbitration

The I/O APIC uses one wire arbitration to win bus ownership. A rotating priority scheme is used for APIC bus arbitration. The winner of the arbitration becomes the lowest priority agent and assumes an arbitration ID of 0. All other agents, except the agent whose arbitration ID is 15, increment their Arbitration IDs by one. The agent whose ID was 15 takes the winner's arbitration ID and increments it by one. Arbitration IDs are changed only for messages that are transmitted successfully (except for the Low Priority messages). A message is transmitted successfully if no CS error or acceptance error was reported for that message.

An APIC agent can use two different priority schemes: Normal or EOI. EOI has the highest priority. EOI priority is used to send EOI messages for level interrupts from a local APIC to an I/O APIC. When an agent requests the bus with EOI priority, all other agents requesting the bus with normal priorities will back off.

When the ICH detects a bus idle condition on the APIC Bus, and it has an interrupt to send over the APIC bus, it drives a start cycle to begin arbitration, by driving bit 0 to a '0' on an APICCLK rising edge. It then samples bit 1. If Bit 1 was a 0, then a local APIC started arbitration for an EOI message on the same clock edge that the ICH started arbitration. The ICH has, thus, lost arbitration and stops driving the APIC bus.

If the ICH did not see an EOI message start, it starts transferring its arbitration ID, located in bits [27:24] of its Arbitration ID register (ARBID). Starting in Cycle 2, through Cycle 5, it tri-states bit 0, and drives bit 1 to a '0' if ARBID[27] is a '1'. If ARBID[27] is a '0', it also tri-states bit 1. At the end of each cycle, the ICH samples the state of Bit 1 on the APIC bus. If the ICH did not drive Bit 1 (ARBID[27] = '0'), and it samples a '0', then another APIC agent started arbitration for the APIC bus at the same time as the ICH, and it has higher priority. The ICH stops driving the APIC bus. Table 5-17 describes the arbitration cycles.

**Table 5-17. Arbitration Cycles**

| Cycle | Bit 1           | Bit 0 | Comment   |
|-------|-----------------|-------|---|
| 1     | EOI             | 0     | Bit 1 = 1: Normal, Bit 1 = 0: EOI   |
| 2     | NOT (ARBID[27]) | 1     | Arbitration ID. If ICH samples a different value than it sent, it lost arbitration. |
| 3     | NOT (ARBID[26]) | 1     |   |
| 4     | NOT (ARBID[25]) | 1     |   |
| 5     | NOT (ARBID[24]) | 1     |   |

### 5.8.3.3 Bus Message Formats

After bus arbitration, the winner is granted exclusive use of the bus and drives its message. APIC messages come in four formats, determined by the delivery mode bits. These four messages are of different length, and are known by all APICs on the bus through the transmission of the Delivery Mode bits:

**Table 5-18. APIC Message Formats**

| Message         | # of Cycles | Delivery Mode Bits      | Comments   |
|-----------------|-------------|-------------------------|--|
| EOI             | 14          | xxx                     | End of Interrupt transmission from Local APIC to I/O APIC on Level interrupts. EOI is known by the EOI bit at the start of arbitration |
| Short           | 21          | 001, 010, 100, 101, 111 | I/O APIC delivery on Fixed, NMI, SMI, Reset, ExtINT, and Lowest Priority with focus processor messages                                 |
| Lowest Priority | 33          | 001                     | Transmission of Lowest Priority interrupts when the status field indicates that the processor doesn't have focus                       |
| Remote Read     | 39          | 011                     | Message from one Local APIC to another to read registers.  |

#### EOI Message For Level Triggered Interrupts

EOI messages are used by local APICs to send an EOI cycle occurring for a level triggered interrupt to an I/O APIC. This message is needed so that the I/O APIC can differentiate between a new interrupt on the interrupt line versus the same interrupt on the interrupt line. The target of the EOI is given by the local APIC through the transmission of the priority vector (V7 through V0) of the interrupt. Upon receiving this message, the I/O APIC resets the Remote IRR bit for that interrupt. If the interrupt signal is still active after the IRR bit is reset, the I/O APIC treats it as a new interrupt.

**Table 5-19. EOI Message**

| Cycle | Bit 1   | Bit 0   | Comments  |
|-------|---------|---------|---|
| 1     | 0       | 0       | EOI message   |
| 2 - 5 | ARBID   | 1       | Arbitration ID  |
| 6     | NOT(V7) | NOT(V6) | Interrupt vector bits V7 - V0 from redirection table register |
| 7     | NOT(V5) | NOT(V4) |   |
| 8     | NOT(V3) | NOT(V2) |   |
| 9     | NOT(V1) | NOT(V0) |   |
| 10    | NOT(C1) | NOT(C0) | Check Sum from Cycles 6 - 9                                   |
| 11    | 1       | 1       | Postamble   |
| 12    | NOT(A)  | NOT(A)  | Status Cycle 0  |
| 13    | NOT(A1) | NOT(A1) | Status Cycle 1  |
| 14    | 1       | 1       | Idle  |

### Short Message

Short messages are used for the delivery of Fixed, NMI, SMI, Reset, ExtINT and Lowest Priority with Focus processor interrupts. The delivery mode bits (M2-M0) specify the message. All short messages take 21 cycles including the idle cycle.

**Table 5-20. Short Message**

| Cycle | Bit 1   | Bit 0   | Comments   |
|-------|---------|---------|--|
| 1     | 1       | 0       | Normal Arbitration   |
| 2 - 5 | ARBID   | 1       | Arbitration ID   |
| 6     | NOT(DM) | NOT(M2) | DM <sup>1</sup> = Destination Mode from bit 11 of the redirection table register |
| 7     | NOT(M1) | NOT(M0) | M2-M0 = Delivery Mode from bits 10:8 of the redirection table register           |
| 8     | NOT(L)  | NOT(TM) | L = Level, TM = Trigger Mode   |
| 9     | NOT(V7) | NOT(V6) | Interrupt vector bits V7 - V0 from redirection table register                    |
| 10    | NOT(V5) | NOT(V4) |  |
| 11    | NOT(V3) | NOT(V2) |  |
| 12    | NOT(V1) | NOT(V0) |  |
| 13    | NOT(D7) | NOT(D6) | Destination field from bits 63:56 of redirection table register <sup>1</sup>     |
| 14    | NOT(D5) | NOT(D4) |  |
| 15    | NOT(D3) | NOT(D2) |  |
| 16    | NOT(D1) | NOT(D0) |  |
| 17    | NOT(C1) | NOT(C0) | Checksum for Cycles 6 - 16 <sup>2</sup>  |
| 18    | 1       | 1       | Postamble <sup>3</sup>   |
| 19    | NOT(A)  | NOT(A)  | Status Cycle 0. See Table 5-21.  |
| 20    | NOT(A1) | NOT(A1) | Status Cycle 1. See Table 5-21.  |
| 21    | 1       | 1       | Idle   |

#### NOTES:

1. If DM is 0 (physical mode), then cycles 15 and 16 are the APIC ID and cycles 13 and 14 are sent as '1'. If DM is 1 (logical mode), then cycles 13 through 16 are the 8-bit Destination field. The interpretation of the logical mode 8-bit Destination field is performed by the local units using the Destination Format Register. Shorthand's of "all-incl-self" and "all-excl-self" both use Physical Destination mode and a destination field containing APIC ID value of all ones. The sending APIC knows whether it should (incl) or should not (excl) respond to its own message.
2. The checksum field is the cumulative add (mod 4) of all data bits (DM, M0-3, L, TM, V0-7, D0-7). The APIC driving the message provides this checksum. This, in essence, is the lower two bits of an adder at the end of the message.
3. This cycle allows all APICs to perform various internal computations based on the information contained in the received message. One of the computations takes the checksum of the data received in cycles 6 through 16 and compares it with the value in cycle 18. If any APIC computes a different checksum than the one passed in cycle 17, then that APIC signals an error on the APIC bus ("00") in cycle 19. If this happens, all APICs will assume the message was never sent and the sender must try sending the message again, which includes re-arbitrating for the APIC bus. In lowest priority delivery when the interrupt has a focus processor, the focus processor signals this by driving a "01" during cycle 19. This tells all the other APICs that the interrupt has been accepted, the arbitration is preempted, and short message format is used. Cycle 19 and 20 indicates the status of the message (i.e., accepted, check sum error, retry or error). Table 5-21 shows the status signal combinations and their meanings for all delivery modes.



Table 5-21. APIC Bus Status Cycle Definition

| Delivery Mode           | A  | Comments                        | A1 | Comments                        |
|-------------------------|----|---------------------------------|----|---------------------------------|
| Fixed, EOI              | 11 | Checksum OK                     | 1x | Error                           |
|                         |    |                                 | 01 | Accepted                        |
|                         |    |                                 | 00 | Retry                           |
|                         | 10 | Error                           | xx |                                 |
|                         | 01 | Error                           | xx |                                 |
|                         | 00 | Checksum Error                  | xx |                                 |
| NMI, SMM, Reset, ExtINT | 11 | Checksum OK                     | 1x | Error                           |
|                         |    |                                 | 01 | Accepted                        |
|                         |    |                                 | 00 | Error                           |
|                         | 10 | Error                           | xx |                                 |
|                         | 01 | Error                           | xx |                                 |
|                         | 00 | Checksum Error                  | xx |                                 |
| Lowest Priority         | 11 | Checksum OK: No Focus Processor | 1x | Error                           |
|                         |    |                                 | 01 | End and Retry                   |
|                         |    |                                 | 00 | Go for Low Priority Arbitration |
|                         | 10 | Error                           | xx |                                 |
|                         | 01 | Checksum OK: Focus Processor    | xx |                                 |
|                         | 00 | Checksum Error                  | xx |                                 |
| Remote Read             | 11 | Checksum OK                     | xx |                                 |
|                         | 10 | Error                           | xx |                                 |
|                         | 01 | Error                           | xx |                                 |
|                         | 00 | Checksum Error                  | xx |                                 |

**Lowest Priority without Focus Processor (FP) Message**

This message format is used to deliver an interrupt in the lowest priority mode in which it does not have a Focus Process. Cycles 1 through 21 for this message is same as for the short message discussed above. Status cycle 19 identifies if there is a Focus processor (10) and a status value of 11 in cycle 20 indicates the need for lowest priority arbitration.

**Table 5-22. Lowest Priority Message (Without Focus Processor)**

| Cycle | Bit 1   | Bit 0   | Comments   |
|-------|---------|---------|--|
| 1     | 1       | 0       | Normal Arbitration   |
| 2 - 5 | ARBID   | 1       | Arbitration ID   |
| 6     | NOT(DM) | NOT(M2) | DM = Destination Mode from bit 11 of the redirection table register    |
| 7     | NOT(M1) | NOT(M0) | M2-M0 = Delivery Mode from bits 10:8 of the redirection table register |
| 8     | NOT(L)  | NOT(TM) | L = Level, TM = Trigger Mode   |
| 9     | NOT(V7) | NOT(V6) | Interrupt vector bits V7 - V0 from redirection table register          |
| 10    | NOT(V5) | NOT(V4) |  |
| 11    | NOT(V3) | NOT(V2) |  |
| 12    | NOT(V1) | NOT(V0) |  |
| 13    | NOT(D7) | NOT(D6) | Destination field from bits 63:56 of redirection table register        |
| 14    | NOT(D5) | NOT(D4) |  |
| 15    | NOT(D3) | NOT(D2) |  |
| 16    | NOT(D1) | NOT(D0) |  |
| 17    | NOT(C1) | NOT(C0) | Checksum for Cycles 6 - 16   |
| 18    | 1       | 1       | Postamble  |
| 19    | NOT(A)  | NOT(A)  | Status Cycle 0.  |
| 20    | NOT(A1) | NOT(A1) | Status Cycle 1.  |
| 21    | P7      | 1       | Inverted Processor Priority P7 - P0                                    |
| 22    | P6      | 1       |  |
| 23    | P5      | 1       |  |
| 24    | P4      | 1       |  |
| 25    | P3      | 1       |  |
| 26    | P2      | 1       |  |
| 27    | P1      | 1       |  |
| 28    | P0      | 1       |  |
| 29    | ArbID3  | 1       |  |
| 30    | ArbID2  | 1       |  |
| 31    | ArbID1  | 1       |  |
| 32    | ArbID0  | 1       |  |
| 33    | S       | S       | Status   |
| 34    | 1       | 1       | Idle   |

**NOTES:**

1. Cycle 21 through 28 are used to arbitrate for the lowest priority processor. The processor that takes part in the arbitration drives the processor priority on the bus. Only the local APICs that have "free interrupt slots" will participate in the lowest priority arbitration.
2. Cycles 29 through 32 are used to break tie in case two more processors have lowest priority. The bus arbitration ID's are used to break the tie.

### Remote Read Message

Remote read message is used when a local APIC wishes to read the register in another local APIC. The message format is same as short message for the first 21 cycles.

**Table 5-23. Remote Read Message**

| Cycle | Bit 1   | Bit 0   | Comments   |
|-------|---------|---------|--|
| 1     | 1       | 0       | Normal Arbitration   |
| 2 - 5 | ARBID   | 1       | Arbitration ID   |
| 6     | NOT(DM) | NOT(M2) | DM = Destination Mode from bit 11 of the redirection table register    |
| 7     | NOT(M1) | NOT(M0) | M2-M0 = Delivery Mode from bits 10:8 of the redirection table register |
| 8     | NOT(L)  | NOT(TM) | L = Level, TM = Trigger Mode   |
| 9     | NOT(V7) | NOT(V6) | Interrupt vector bits V7 - V0 from redirection table register          |
| 10    | NOT(V5) | NOT(V4) |  |
| 11    | NOT(V3) | NOT(V2) |  |
| 12    | NOT(V1) | NOT(V0) |  |
| 13    | NOT(D7) | NOT(D6) | Destination field from bits 63:56 of redirection table register        |
| 14    | NOT(D5) | NOT(D4) |  |
| 15    | NOT(D3) | NOT(D2) |  |
| 16    | NOT(D1) | NOT(D0) |  |
| 17    | NOT(C1) | NOT(C0) | Checksum for Cycles 6 - 16   |
| 18    | 1       | 1       | Postamble  |
| 19    | NOT(A)  | NOT(A)  | Status Cycle 0.  |
| 20    | NOT(A1) | NOT(A1) | Status Cycle 1.  |
| 21    | d31     | d30     | Remote register data 31-0  |
| 22    | d29     | d28     |  |
| 23    | d27     | d26     |  |
| 24    | d25     | d24     |  |
| 25    | d23     | d22     |  |
| 26    | d21     | d20     |  |
| 27    | d19     | d18     |  |
| 28    | d17     | d16     |  |
| 29    | d15     | d14     |  |
| 30    | d13     | d12     |  |
| 31    | d11     | d10     |  |
| 32    | d09     | d08     |  |
| 33    | d07     | d06     |  |
| 34    | d05     | d04     |  |
| 35    | d03     | d02     |  |
| 36    | d01     | d00     |  |
| 37    | S       | S       | Data Status: 00 = valid, 11 = invalid                                  |
| 38    | C       | C       | Check Sum for data d31-d00   |
| 39    | 1       | 1       | Idle   |

**NOTE:** Cycle 21 through 36 contain the remote register data. The status information in cycle 37 specifies if the data is good or not. Remote read cycle is always successful (although the data may be valid or invalid) in that it is never retried. The reason for this is that Remote Read is a debug feature, and a "hung" remote APIC that is unable to respond should not cause the debugger to hang.

## 5.9 Serial Interrupt (D31:F0)

The ICH supports a serial IRQ scheme. This allows a single signal to be used to report interrupt requests. The signal used to transmit this information is shared between the host, the ICH, and all peripherals that support serial interrupts. The signal line, SERIRQ, is synchronous to PCI clock, and follows the sustained tri-state protocol that is used by all PCI signals. This means that if a device has driven SERIRQ low, it first drives it high synchronous to PCI clock and releases it the following PCI clock. The serial IRQ protocol defines this sustained tri-state signaling in the following fashion:

- **S - Sample Phase.** Signal driven low
- **R - Recovery Phase.** Signal driven high
- **T - Turn-around Phase.** Signal released

The ICH supports a message for 21 serial interrupts. These represent the 15 ISA interrupts (IRQ0-1, 2-15), the four PCI interrupts, and the control signals SMI# and IOCHK#. The serial IRQ protocol does not support the additional APIC interrupts (20-23).

### 5.9.1 Start Frame

The serial IRQ protocol has two modes of operation which affect the start frame. These two modes are: Continuous, where the ICH is solely responsible for generating the start frame; and Quiet, where a serial IRQ peripheral is responsible for beginning the start frame.

The mode that must first be entered when enabling the serial IRQ protocol is continuous mode. In this mode, the ICH asserts the start frame. This start frame is 4, 6, or 8 PCI clocks wide based upon the Serial IRQ Control Register, bits 1:0 at 64h in Device 31:Function 0 configuration space. This is a polling mode.

When the serial IRQ stream enters quiet mode (signaled in the Stop Frame), the SERIRQ line remains inactive and pulled up between the Stop and Start Frame until a peripheral drives the SERIRQ signal low. The ICH senses the line low and continues to drive it low for the remainder of the Start Frame. Since the first PCI clock of the start frame was driven by the peripheral in this mode, the ICH drives the SERIRQ line low for 1 PCI clock less than in continuous mode. This mode of operation allows for a quiet, and therefore lower power, operation.

### 5.9.2 Data Frames

Once the Start frame has been initiated, all of the SERIRQ peripherals must start counting frames based on the rising edge of SERIRQ. Each of the IRQ/DATA frames has exactly 3 phases of 1 clock each:

- **Sample Phase.** During this phase, the SERIRQ device drives SERIRQ low if the corresponding interrupt signal is low. If the corresponding interrupt is high, then the SERIRQ devices tri-state the SERIRQ signal. The SERIRQ line remains high due to pull-up resistors.
- **Recovery Phase.** During this phase, the device drives the SERIRQ line high if in the Sample Phase it was driven low. If it was not driven in the sample phase, it is tri-stated in this phase.
- **Turn-around Phase.** The device tri-states the SERIRQ line

### 5.9.3 Stop Frame

After all data frames, a Stop Frame is driven by the ICH. The SERIRQ signal is driven low by the ICH for 2 or 3 PCI clocks. The number of clocks is determined by the SERIRQ configuration register. The number of clocks determines the next mode:

Table 5-24. Stop Frame Explanation

| Stop Frame Width | Next Mode  |
|------------------|--|
| 2 PCI clocks     | <b>Quiet Mode.</b> Any SERIRQ device may initiate a Start Frame        |
| 3 PCI clocks     | <b>Continuous Mode.</b> Only the host (ICH) may initiate a Start Frame |

### 5.9.4 Specific Interrupts not Supported via SERIRQ

There are three interrupts seen through the serial stream which are not supported by the ICH. These interrupts are generated internally, and are not sharable with other devices within the system. These interrupts are:

- IRQ0. Heartbeat interrupt generated off of the internal 8254 counter 0.
- IRQ8#. RTC interrupt can only be generated internally.
- IRQ13. Floating point error interrupt generated off of the processor assertion of FERR#.

The ICH ignores the state of these interrupts in the serial stream, and does not adjust their level based on the level seen in the serial stream. In addition, the interrupts IRQ14 and IRQ15 from the serial stream are treated differently than their ISA counterparts. These two frames are not passed to the Bus Master IDE logic. The Bus Master IDE logic expects IDE to be behind the ICH.

## 5.9.5 Data Frame Format

Table 5-25 shows the format of the data frames. For the PCI interrupts (A-D), the output from the ICH is ANDed with the PCI input signal. This way, the interrupt can be signaled via both the PCI interrupt input signal and via the SERIRQ signal (they are shared).

Table 5-25. Data Frame Format

| Data Frame # | Interrupt | Clocks Past Start Frame | Comment  |
|--------------|-----------|-------------------------|--|
| 1            | IRQ0      | 2                       | Ignored. IRQ0 can only be generated via the internal 8524  |
| 2            | IRQ1      | 5                       |  |
| 3            | SMI#      | 8                       | Causes SMI# if low. Will set SERIRQ_SMI_STS status bit.    |
| 4            | IRQ3      | 11                      |  |
| 5            | IRQ4      | 14                      |  |
| 6            | IRQ5      | 17                      |  |
| 7            | IRQ6      | 20                      |  |
| 8            | IRQ7      | 23                      |  |
| 9            | IRQ8      | 26                      | Ignored. IRQ8# can only be generated internally or on ISA. |
| 10           | IRQ9      | 29                      |  |
| 11           | IRQ10     | 32                      |  |
| 12           | IRQ11     | 35                      |  |
| 13           | IRQ12     | 38                      |  |
| 14           | IRQ13     | 41                      | Ignored. IRQ13 can only be generated from FERR#            |
| 15           | IRQ14     | 44                      | Do not include in BM IDE interrupt logic                   |
| 16           | IRQ15     | 47                      | Do not include in BM IDE interrupt logic                   |
| 17           | IOCHCK#   | 50                      | Same as ISA IOCHCK# going active.                          |
| 18           | PCI INTA# | 53                      | Drive PIRQA#   |
| 19           | PCI INTB# | 56                      | Drive PIRQB#   |
| 20           | PCI INTC# | 59                      | Drive PIRQC#   |
| 21           | PCI INTD# | 62                      | Drive PIRQD#   |

## 5.10 Real Time Clock (D31:F0)

The Real Time Clock (RTC) module provides a battery backed-up date and time keeping device with two banks of static RAM with 128 bytes each, although the first bank has 114 bytes for general purpose usage. Three interrupt features are available: time of day alarm with once a second to once a month range, periodic rates of 122us to 500ms, and end of update cycle notification. Seconds, minutes, hours, days, day of week, month, and year are counted. Daylight savings compensation is optional. The hour is represented in twelve or twenty-four hour format, and data can be represented in BCD or binary format. The design is meant to be functionally compatible with the Motorola MS146818B. The time keeping comes from a 32.768 kHz oscillating source, which is divided to achieve an update every second. The lower 14 bytes on the lower RAM block has very specific functions. The first ten are for time and date information. The next four (0Ah to 0Dh) are registers, which configure and report RTC functions.

The time and calendar data should match the data mode (BCD or binary) and hour mode (12 or 24 hour) as selected in register B. It is up to the programmer to make sure that data stored in these locations is within the reasonable values ranges and represents a possible date and time. The exception to these ranges is to store a value of C0 - FF in the Alarm bytes to indicate a don't care situation. All Alarm conditions must match to trigger an Alarm Flag, which could trigger an Alarm Interrupt if enabled. The SET bit of register B should be one while programming these locations to avoid clashes with an update cycle. Access to time and date information is done through the RAM locations. If a RAM read from the ten time and date bytes is attempted during an update cycle, the value read does not necessarily represent the true contents of those locations. Any RAM writes under the same conditions are ignored.

**Note:** The ICH supports the ability to generate an SMI# based on Year 2000 rollover. See Section 5.10.4 for more information on the century rollover.

The ICH does not implement month/year alarms.

### 5.10.1 Update Cycles

An update cycle occurs once a second, if the SET bit of register B is not asserted and the divide chain is properly configured. During this procedure, the stored time and date are incremented, overflow checked, a matching alarm condition checked, and the time and date rewritten to the RAM locations. To maintain compatibility with the Motorola MS146818B, the update cycle starts at least 244us after the UIP bit of register A is asserted, and the entire cycle will not take more than 1984us to complete. The time and date RAM locations (0-9) will be disconnected from the external bus during this time.

To avoid update and data corruption conditions, external RAM access to these locations can safely occur at two times. When a updated-ended interrupt is detected, almost 999ms is available to read and write the valid time and date data. If the UIP bit of Register A is detected to be low, there is at least 244us before the update cycle begins.

**Warning:** The overflow conditions for leap years and daylight savings adjustments are based on more than one date or time item. To ensure proper operation when adjusting the time, the new time and data values should be set at least two seconds before one of these conditions (leap year, daylight savings time adjustments) occurs.

## 5.10.2 Interrupts

The real-time clock interrupt is internally routed within the ICH both to the I/O APIC and the 8259. It is mapped to interrupt vector 8. This interrupt does not leave the ICH, nor is it shared with any other interrupt. IRQ8# from the SERIRQ stream is ignored.

## 5.10.3 Lockable RAM Ranges

The RTC's battery-backed RAM supports two 8-byte ranges that can be locked via the configuration space. If the locking bits are set, the corresponding range in the RAM will not be readable or writable. A write cycle to those locations have no effect. A read cycle to those locations does not return the location's actual value (may be all 0's or all 1's).

Once a range is locked, the range can be unlocked only by a hard reset, which invokes the BIOS and allows it to relock the RAM range.

## 5.10.4 Century Rollover

The ICH detects a rollover when the Year byte (RTC I/O space, index offset 09h) transitions from 99 to 00. Upon detecting the rollover, the ICH sets the NEWCENTURY\_STS bit (TCOBASE + 04h, bit 7). If the system is in an S0 state, this causes an SMI#. The SMI# handler can update registers in the RTC RAM that are associated with century value. If the system is in a sleep state (S1-S5) when the century rollover occurs, the ICH also sets the NEWCENTURY\_STS bit, but no SMI# is generated. When the system resumes from the sleep state, BIOS should check the NEWCENTURY\_STS bit and update the century value in the RTC RAM.

## 5.10.5 Clearing Battery-Backed RTC RAM

Clearing CMOS RAM in an ICH-based platform can be done by using a jumper on RTCRST# or GPI, or using SAFEMODE strap. Implementations should not attempt to clear CMOS by using a jumper to pull VccRTC low.



### Using RTCRST# to clear CMOS

A jumper on RTCRST# can be used to clear CMOS values, as well as reset to default, the state of those configuration bits that reside in the RTC power well. When the RTCRST# is strapped to ground, the RTC\_PWR\_STS bit (D31:F0:A4h bit 2) is set and those configuration bits in the RTC power well are set to their default state. BIOS can monitor the state of this bit, and manually clear the RTC CMOS array once the system is booted. The normal position would cause RTCRST# to be pulled up through a weak pull-up resistor. The following table shows which bits are set to their default state when RTCRST# is asserted

**Table 5-26. Configuration Bits Reset By RTCRST# Assertion**

| Bit Name                          | Default State  | Register                       | Location           | Bit(s)  |
|-----------------------------------|----------------|--------------------------------|--------------------|---------|
| CPU_BIST_EN<br>(ICH 82801AA only) | 0              | General Status                 | D31:F0:D7-D4       | 12      |
| FREQ_STRAP[3:0]                   | 0Fh            | General Status                 | D31:F0:D7-D4       | 11:8    |
|                                   | undefined      | RTC Register A                 | I/O space          | ALL     |
| PIE, UIE, SQWE                    | 0, 0, 0        | RTC Register B                 | I/O space          | 6,4,3   |
| IRQF, PF, AF, UF                  | 0, 0, undef, 0 | RTC Register C                 | I/O space          | 7,6,5,4 |
| VRT, Date Alarm                   | 1, undefined   | RTC Register D                 | I/O space          | 7, 5:0  |
| RTC_PWR_STS                       | 0              | General PM Configuration 3     | D31:F0:A4          | 2       |
| PWR_FLR                           | 0              | General PM Configuration 3     | D31:F0:A4          | 1       |
| AFTERG3_EN                        | 0              | General PM Configuration 3     | D31:F0:A4          | 0       |
| PRBTNOR_STS                       | 0              | Power Management 1 Status      | D31:F0 PMBase+0h   | 11      |
| PME_EN, RI_STS                    | 0, 0           | General Purpose Event 0 Enable | D31:F0 PMBase+2Ah  | 11,8    |
| NEW_CENTURY_STS                   | 0              | TCO1 Status                    | D31:F0 TCOBase+04h | 7       |
| INTRD_DET                         | 0              | TCO2 Status                    | D31:F0 TCOBase+06h | 0       |

### Using a GPI to Clear CMOS

A jumper on a GPI can also be used to clear CMOS values. BIOS would detect the setting of this GPI on system boot-up, and manually clear the CMOS array.

### Using the SAFEMODE Strap to Clear CMOS

A jumper on AC\_SDOOUT (SAFEMODE strap) can also be used to clear CMOS values. BIOS would detect the setting of the SAFE\_MODE status bit (D31:F0: Offset D4h bit 2) on system boot-up, and manually clear the CMOS array.

**Note:** Both the GPI & SAFEMODE strap techniques to clear CMOS require multiple steps to implement. The system is booted with the jumper in new position, then powered back down. The jumper is replaced back to the normal position, then the system is rebooted again. The RTCRST# jumper technique allows the jumper to be moved and then replaced, all while the system is powered off. Then, once booted, the RTC\_PWR\_STS can be detected in the set state.

**Note:** Clearing CMOS, using a jumper on VCCRTC, must NOT be implemented.

## 5.11 Processor Interface (D31:F0)

The ICH has the following outputs to the processor:

- A20M#, SMI#, NMI, INIT#, INTR, STPCLK#, IGNNE#, CPUSLP#

The ICH outputs to the processor use open drain buffers, which are pulled up at the system level to the processor CMOS I/O buffer voltage. The ICH contains one input from the processor, FERR#, which has special buffer requirements for the different voltage levels. The  $V_{il}$  threshold needs to be compatible with processors that will not drive the signal above 1.8V.

The ICH also handles the speed setting for the processor by holding specific signals at certain states just prior to CPURST going inactive. This avoids the glue often required with other PCIsets. The ICH does not attempt to support the processor's FRC mode, as this changes the behavior of many signals.

### 5.11.1 Processor Interface Signals

This section describes each of the signals that interface between the ICH and the processor bus. Note that the behavior of some signals may vary during processor reset, as the signals are used for frequency strapping.

#### A20M#

The A20M# signal will be active (low) when both of the following conditions are true:

- The ALT\_A20\_GATE bit (Bit 1 of PORT92 register) is a '0'
- The A20GATE input signal is a '0'

The A20GATE input signal is expected to be generated by the external microcontroller (KBC).

#### INIT#

The INIT# signal will be active (driven low) based on any one of several events described in Table 5-27. When any of these events occur, INIT# will be driven low for 16 PCI clocks, then released (and pulled high by external pull-up resistor).

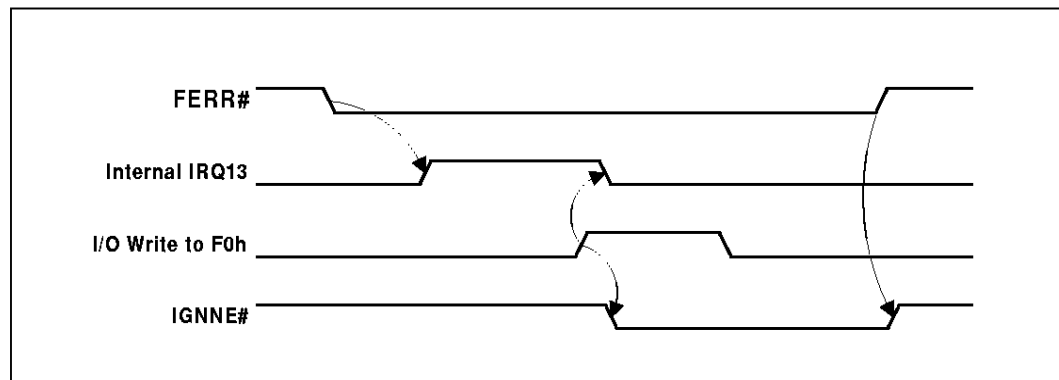
**Note:** The 16-clock counter for INIT# assertion will halt while STPCLK# is active. Therefore, if INIT# is supposed to go active while STPCLK# is asserted, it will actually go active after STPCLK# goes inactive.

**Table 5-27. INIT# Going Active**

| Cause of INIT# Going Active  | Comment   |
|--|---|
| Shutdown special cycle from processor.   |   |
| PORT92 write, where INIT_NOW (bit 0) transitions from a 0 to a 1.                                  |   |
| PORTCF9 write, where RST_CPU (bit 2) was a 0 and SYS_RST(bit 1) transitions from 0 to 1.           |   |
| RCIN# input signal goes low. RCIN# is expected to be driven by the external microcontroller (KBC). | 0 to 1 transition on RCIN# must occur before the ICH will arm INIT# to be generated again.                    |
| Processor BIST (ICH: 82801AA only)   | To enter BIST, the software sets CPU_BIST_EN bit and then does a full processor reset using the CF9 register. |

### FERR#/IGNNE# (Coprocessor Error)

The ICH supports the coprocessor error function with the FERR#/IGNNE# pins. The function is enabled via the COPROC\_ERR\_EN bit (Device 31:Function 0, Offset D0, bit 13). FERR# is tied directly to the Coprocessor Error signal of the processor. If FERR# is driven active by the processor, IRQ13 goes active (internally). When it detects a write to the COPROC\_ERR register, the ICH negates the internal IRQ13 and drives IGNNE# active. IGNNE# remains active until FERR# is driven inactive. IGNNE# is never driven active unless FERR# is active.

**Figure 5-10. Coprocessor Error Timing Diagram**


If COPROC\_ERR\_EN is not set, then the assertion of FERR# will have not generate an internal IRQ13, nor will the write to F0h generate IGNNE#.

### NMI

Non-Maskable Interrupts (NMIs) can be generated by several sources.

**Table 5-28. NMI Sources**

| Cause of NMI  | Comment   |
|---|---|
| SERR# goes active (either internally, externally via SERR# signal, or via message from the host controller) | Can instead be routed to generate an SCI, through the NMI2SCI_EN bit (Device 31:Function 0, offset 4E, bit 11). |
| IOCHK# goes active via SERIRQ# stream (ISA system Error)  | Can instead be routed to generate an SCI, through the NMI2SCI_EN bit (Device 31:Function 0, offset 4E, bit 11). |

### STPCLK# and CPUSLP# Signals

The ICH power management logic controls these active-low open-drain signals. Refer to Section 5.1.2 for more information on the functionality of these signals.

## 5.11.2 Dual Processor Issues (ICH: 82801AA only)

### 5.11.2.1 Signal Differences (ICH: 82801AA only)

In dual processor designs, some of the processor signals are unused or used differently than for uniprocessor designs.

Table 5-29. DP Signal Differences

| Signal          | Difference  |
|-----------------|---|
| A20M# / A20GATE | Generally not used, but still supported by ICH.   |
| STPCLK#         | Used for S1 State as well as preparation for entry to S3-S5<br>Also allows for THERM# based throttling (not via ACPI control methods).<br>Should be connected to both processors. |
| FERR# / IGNNE#  | Generally not used, but still supported by ICH.   |

### 5.11.2.2 Power Management (ICH: 82801AA only)

Attempting clock control with more than one processor is not feasible, because the MCH does not provide any indication as to which processor is executing a particular Stop-Grant cycle. Without this information, there is no way for the 82801AA ICH to know when it is safe to deassert STPCLK#.

Because the S1 state will have the STPCLK# signal active, the STPCLK# signal can be connected to both processors. However, for ACPI implementations, the ICH will not support the C2 state for both processors, since there are not two processor control blocks. The BIOS must indicate that the ICH only supports the C1 state for dual processor designs. However, the THERM# signal can be used for overheat conditions to activate thermal throttling.

When entering S1, the ICH asserts STPCLK# to both processors. The ICH will then wait 8 PCI clocks after receipt of the first Stop-Grant Acknowledge cycle before asserting CPUSLP# (if the SLP\_EN bit is set to 1).

Both processors must immediately respond to the STPCLK# assertion with stop grant acknowledge cycles before the ICH asserts CPUSLP# in order to meet the processor setup time for CPUSLP#. Meeting the processor setup time for CPUSLP# is not an issue if both processors are idle when the system is entering S1. If you cannot guarantee that both processors will be idle, do not enable the SLP\_EN bit. Note that setting SLP\_EN to 1 is not required to support S1 in a dual processor configuration.

S2 is not supported for single or dual processor desktop designs.

In going to the S3, S4, or S5 states, the system will appear to pass through the S1 state, and thus STPCLK# and SLP# are also used.

During the S3, S4, and S5 states, both processors will lose power. Upon exit from those states, the processors will have their power restored.

### 5.11.3 Speed Strapping for Processor

The ICH directly sets the speed straps for the processor, saving the external logic that has been needed with prior PCIsets. Refer to processor specification for speed strapping definition.

The ICH will perform the following to set the speed straps for the processor:

1. While PCIRST# is active, the ICH will drive A20M#, IGNNE#, NMI, and INTR high.
2. As soon as PWROK goes active, the ICH reads the `FREQ_STRAP` field contents.
3. The next step depends on the power state being exited as described in Table 5-30.

**Table 5-30. Frequency Strap Behavior Based on Exit State**

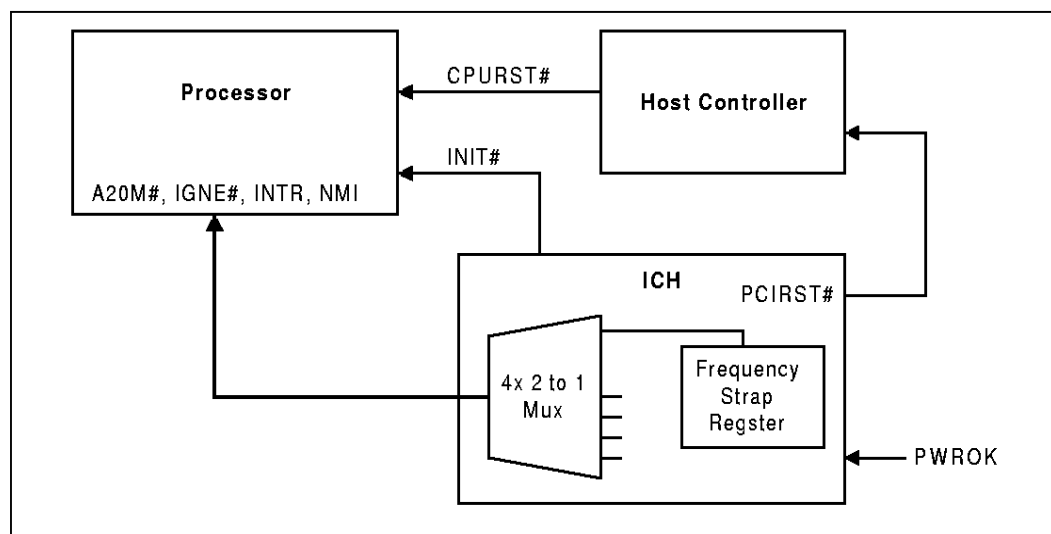
| State Exiting     | ICH   |
|-------------------|---|
| S1                | There is no processor reset; Thus, there is no frequency strap logic is used.   |
| S3, S4, S5, or G3 | Based on PWROK going active, the ICH will deassert PCIRST#, and based on the value of the <code>FREQ_STRAP</code> field (D31:F0, Offset D4), the ICH will drive the intended core frequency values on A20M#, IGNNE#, NMI, and INTR. The ICH will hold these signals for 120ns after CPURST# is deasserted by the host controller. |

**Table 5-31. Frequency Strap Bit Mapping**

| <code>FREQ_STRAP</code> bits [3:0] | Sets High/Low level for corresponding signal |
|------------------------------------|--|
| 3                                  | NMI  |
| 2                                  | INTR   |
| 1                                  | IGNNE#                                       |
| 0                                  | A20M#  |

**NOTE:** The `FREQ_STRAP` register is in the RTC well. The value in the register can be forced to 1111h via a pinstrap (`AC_SDOUT` signal), or the ICH can automatically force the speed strapping to 1111h if the processor fails to boot.

**Figure 5-11. ICH Signal Strapping**



## 5.12 Power Management (D31:F0)

### 5.12.1 ICH and System Power States

Table 5-32 shows the power states defined for ICH-based platforms. The state names generally match the corresponding ACPI states.

**Table 5-32. General Power States for Systems using ICH**

| State/<br>Substates | Legacy Name / Description   |
|---------------------|---|
| G0/S0/C0            | <b>Full On:</b> Processor operating. Individual devices may be shut down to save power. The different processor operating levels are defined by Cx states, as shown in Table 5-33. Within the C0 state, the ICH can throttle the STPCLK# signal to reduce power consumption. The throttling can be initiated by software or by the THRM# input signal.  |
| G0/S0/C1            | <b>Auto-Halt:</b> Processor has executed a AutoHalt instruction and is not executing code. The processor snoops the bus and maintains cache coherency.  |
| G0/S0/C2            | <b>Stop-Grant:</b> The STPCLK# signal goes active to the processor. The processor performs a Stop-Grant cycle, halts its instruction stream, and remain in that state until the STPCLK# signal goes inactive. In the Stop-Grant state, the processor snoops the bus and maintains cache coherency.  |
| G1/S1               | <b>Stop-Grant:</b> Similar to G0/S0/C2 state. ICH also has the option to assert the processor SLP# signal to further reduce processor power consumption.  |
| G1/S3               | <b>Suspend-To-RAM (STR):</b> The system context is maintained in system DRAM, but power is shut off to non-critical circuits. Memory is retained, and refreshes continue. All clocks stop except RTC clock.   |
| G1/S4               | <b>Suspend-To-Disk (STD):</b> The context of the system is maintained on the disk. All power is then shut off to the system except for the logic required to resume. Externally appears same as S5, but may have different wake events.   |
| G2/S5               | <b>Soft Off (SOFF):</b> System context is not maintained. All power is shut off except for the logic required to restart. A full boot is required when waking.  |
| G3                  | <b>Mechanical OFF (MOFF):</b> System context not maintained. All power is shut off except for the RTC. No "Wake" events are possible, because the system does not have any power. This state occurs if the user removes the batteries, turns off a mechanical switch, or if the system power supply is at a level that is insufficient to power the "waking" logic. When system power returns, transition will depends on the state just prior to the entry to G3 and the AFTERG3 bit in the GEN_PMCON3 register (D31:F0, offset A4). Refer to Table 5-41 for more details. |

Table 5-33 shows the transitions rules among the various states. Note that transitions among the various states may appear to temporarily transition through intermediate states. For example, in going from S0 to S1, it may appear to pass through the G0/S0/C2 states. These intermediate transitions and states are not listed in the table.

**Table 5-33. State Transition Rules for ICH**

| Present State          | Transition Trigger  | Next State  |
|------------------------|---|---|
| G0/S0/C0               | <ul style="list-style-type: none"> <li>Processor halt instruction</li> <li>Level 2 Read</li> <li>SLP_EN bit set</li> <li>Power Button Override</li> <li>Mechanical Off/Power Failure</li> </ul> | <ul style="list-style-type: none"> <li>G0/S0/C1</li> <li>G0/S0/C2</li> <li>G1/Sx or G2/S5state</li> <li>G2/S5</li> <li>G3</li> </ul>            |
| G0/S0/C1               | <ul style="list-style-type: none"> <li>Any Enabled Break Event</li> <li>STPCLK# goes active</li> <li>Power Button Override</li> <li>Power Failure</li> </ul>                                    | <ul style="list-style-type: none"> <li>G0/S0/C0</li> <li>G0/S0/C2</li> <li>G2/S5</li> <li>G3</li> </ul>   |
| G0/S0/C2               | <ul style="list-style-type: none"> <li>Any Enabled Break Event</li> <li>STPCLK# goes inactive and previously in C1</li> <li>Power Button Override</li> <li>Power Failure</li> </ul>             | <ul style="list-style-type: none"> <li>G0/S0/C0</li> <li>G0/S0/C1</li> <li>G2/S5</li> <li>G3</li> </ul>   |
| G1/S1, G1/S3, or G1/S4 | <ul style="list-style-type: none"> <li>Any Enabled Wake Event</li> <li>Power Button Override</li> <li>Power Failure</li> </ul>  | <ul style="list-style-type: none"> <li>G0/S0/C0</li> <li>G2/S5</li> <li>G3</li> </ul>   |
| G2/S5                  | <ul style="list-style-type: none"> <li>Any Enabled Wake Event</li> <li>Power Failure</li> </ul>   | <ul style="list-style-type: none"> <li>G0/S0/C0</li> <li>G3</li> </ul>  |
| G3                     | <ul style="list-style-type: none"> <li>Power Returns</li> </ul>   | <ul style="list-style-type: none"> <li>Optional to go to S0/C0 (reboot) or G2/S5 (stay off until power button pressed). (See Note 1)</li> </ul> |

**NOTE:** Some wake events can be preserved through power failure.

## 5.12.2 System Power Planes

The system has several independent power planes, as described in Table 5-34. Note that when a particular power plane is shut off, it should go to a 0V level.

**Table 5-34. System Power Plane**

| Plane     | Controlled By  | Description  |
|-----------|----------------|--|
| MAIN      | SLP_S3# signal | When SLP_S3# goes active, power can be shut off to any circuit not required to wake the system from the S3 state. Since the S3 state requires that the memory context be preserved, power must be retained to the main memory.<br>Devices on the PCI bus, LPC interface downstream Hub Interface (and AGP for the ICH 82801AA) will, typically, be shut off when the Main power plane is shut; there may be small subsections powered. |
| MEMORY    | SLP_S5# signal | When the SLP_S5# goes active, power can be shut off to any circuit not required to wake the system from the S4 or S5 state. Since the memory context does not need to be preserved in the S5 state, the power to the memory can also be shut down.   |
| DEVICE[n] | GPIO           | Individual subsystems may have their own power plane. For example, GPIO signals may be used to control the power to disk drives, audio amplifiers, or the display screen.  |

### 5.12.3 SMI#/SCI Generation

Upon any SMI# event taking place, the ICH asserts SMI# to the processor, which causes it to enter SMM space. SMI# remains active until the EOS bit is set. When the EOS bit is set, SMI# goes inactive for a minimum of 4 PCICLK. If another SMI event occurs, SMI# is driven active again.

The SCI is a level-mode interrupt that is typically handled by an ACPI-aware operating system. In non-APIC systems (which is the default), the SCI IRQ is routed to one of the 8259 interrupts (IRQ 9, 10, or 11). The 8259 interrupt controller must be programmed to level mode for that interrupt.

In systems using the APIC, the SCI can still be routed to IRQ9, 10, or 11, or it can be instead routed to one of the APIC interrupts 20-23. In either case, the interrupt generated internally is active low level. The interrupt will remain low until all SCI sources are removed.

Table 5-35 shows which events can cause an SMI# and SCI. Note that some events can be programmed to cause either an SMI# or SCI. The usage of the event for SCI (instead of SMI#) is typically associated with an ACPI-based system. Each SMI# or SCI source has a corresponding enable and status bit.

**Table 5-35. Causes of SMI# and/or SCI**

| Event   | SCI | SMI# | Comment  |
|---|-----|------|--|
| Setting of the BIOS_STS bit   |     | X    | ACPI code in OS sets GBL_RLS bit to cause BIOS_STS bit active, which causes SMI#.  |
| Setting of the LEGACY_USB_STS bit   |     | X    | Bit set based on address decode or incoming USB IRQ.   |
| Write access to APM control register  |     | X    | OS or BIOS writes to the APMC register. SMM handler clears.  |
| SW SMI# Timer reaches 0   |     | X    | Allows SMM handler to exit temporarily. Another SMI# occurs about 64 ms later.   |
| Device Trap   |     | X    | Indicates that subsystems may need to be powered back on.  |
| TCO Event (Includes SERR#)  | X   | X    | Can also cause IRQ (other than SCI).   |
| Setting bits in GPE Status Register with corresponding Enable bit set.                    | X   | X    | Bits in GPE Status Register include SMBus, TCO, AC97, RI, PME, USB, THRM and GPIO[n].  |
| Setting of the GBL_STS bit  | X   |      | This bit is set when the BIOS sets the BIOS_RLS bit. The ACPI handler will clear the bit.  |
| Setting any bit in the PM1_STS register with corresponding _EN bit set in the PM1_EN bit. | X   | X    | The SCI/SMI# should be generated, even if waking.  |
| GPIO[n]   | X   | X    | If GPIO programmed as input and individually enabled, Can be Wake event.   |
| Overflow of Power Management Timer  | X   | X    | Time-out every 2.34 seconds. If timer is enabled, will cause an SCI if SCI_EN is set, or it will cause an SMI# if SCI_EN is not set. |
| THRM# signal  | X   | X    | The THRM# can cause an SMI# or SCI on either the rising or falling edge. Causes SCI if SCI_EN is set, causes SMI# if SCI_EN not set. |
| PWRBTN signal   | X   | X    | Can also cause Wake Event  |
| PCI PME# signal   | X   | X    | Can also cause Wake Event  |
| Year 2000 Rollover  |     | X    | Does not cause a Wake Event  |



## 5.12.4 Dynamic System Clock Control

The ICH has extensive control for dynamically starting and stopping system clocks. The clock control is used for transitions among the various S0/Cx states, and processor throttling. Each dynamic clock control method is described in this section. The various Sleep states may also perform types of non-dynamic clock control.

The ICH supports the ACPI C0, C1, C2 states. The Dynamic Clock control is handled using the STPCLK# to halt processor instruction stream. A C2 state ends due to a Break event. Based on the break event, the ICH returns the system to C0 state. Table 5-36 lists the possible break events.

Table 5-36. Break Events

| Event   | Comment  |
|---|--|
| Any unmasked interrupt goes active                    | IRQ[0:15] when using the 8259s, IRQ[0:23] for I/O APIC. Since SCI is an interrupt, any SCI will also be a break event. |
| Any internal event that will cause an NMI or SMI#     | Many possible sources  |
| Any internal event that will cause INIT# to go active | Could be indicated by the keyboard controller via the RCIN input signal.   |

### 5.12.4.1 Entering/Exiting the C1 State

Causes of C1 Entry:

- From C0 State: Processor performs an Autohalt instruction.
- From C2 State: STPCLK# goes inactive and the processor had done an Autohalt instruction prior to STPCLK# going active, and no break event has occurred. The C1 state will persist until an interrupt (regular interrupt, NMI, SMI# , or INIT#) occurs. This C1 state will persist only for a few clocks, since the cause of STPCLK# going inactive is some type of interrupt.

Causes of Exit from C1

- To C0 state: When an enabled interrupt (regular interrupt, NMI, SMI#, or INIT#) occurs
- To C2 state: STPCLK# goes active. This will cause transition to the C2 state. This case can only occur if the processor reads the Level 2 register (forcing a transition to C2 state), and the processor does an autohalt instruction before it recognizes STPCLK# active.

#### 5.12.4.2 Entering/Exiting the C2 State

Causes of S0/C2 Entry:

- From C0 state: Software reads the Level 2 Register
- From C1 state: STPCLK# goes active. This can only occur if the processor does an autohalt instruction just after reading the Level 2 register (so that the autohalt instruction completes prior to the processor recognizing the STPCLK#).

Note that in going to some of the lower power states, the system may appear to pass through the C2 state.

Causes of S0/C2 Exit:

- To C0 state: A Break event occurs. The transition back to the C0 state will be qualified by the Stop-Grant cycle being observed by the ICH. If the break event occurs prior to the Stop-Grant cycle (associated with the prior assertion of STPCLK#), STPCLK# will stay active until the Stop-Grant cycle is observed.
- To C1 state: If STPCLK# goes inactive. The C1 state will persist until an interrupt (regular interrupt, NMI, SMI#, or INIT#) occurs.

#### 5.12.4.3 Throttling Using STPCLK#

Throttling is used to lower power consumption or reduce heat. The ICH asserts STPCLK# to throttle the processor clock and the processor appears to temporarily enter a C2 state. After a programmable time, the ICH deasserts STPCLK# and the processor appears to return to the C0 state. This allows the processor to operate at reduced average power, with a corresponding decrease in performance. Two methods are included to start throttling:

1. Software enables a timer with a programmable duty cycle. The duty cycle is set by the THTL\_DTY field and the throttling is enabled using the THTL\_EN field. This is known as Manual Throttling. The period is fixed to be in the non-audible range, due to the nature of switching power supplies.
2. A Thermal Override condition (THRM# signal active for >2 seconds) occurs that unconditionally forces throttling, independent of the THTL\_EN bit. The throttling due to Thermal Override has a separate duty cycle (THRM\_DTY) which may vary by field and system. The Thermal Override condition will end when THRM# goes inactive.

Throttling due to the THRM# signal has higher priority than the software initiated throttling. Throttling does not occur when the system is in C2 state, even if Thermal override occurs.

#### 5.12.4.4 Transition Rules Among S0/Cx and Throttling States

The following priority rules and assumptions apply among the various S0/Cx and throttling states:

- Entry to any S0/Cx state is mutually exclusive with entry to any S1-S5 state. This is because the processor can only perform one register access at a time and Sleep states have higher priority than thermal throttling.
- When the SLP\_EN bit is set (system going to a sleep state (S1-S5), the THTL\_EN bit can be internally treated as being disabled (no throttling while going to sleep state). Note that thermal throttling (based on THRM# signal) cannot be disabled in an S0 state. However, once the SLP\_EN bit is set, the thermal throttling is shut off (since STPCLK# will be active in S1-S5 states).
- If the THTL\_EN bit is set, and a Level 2 read then occurs, the system should immediately go and stay in a C2 state until a break event occurs. A Level 2 read has higher priority than the software initiated throttling or thermal throttling. If Thermal Override is causing throttling, and a Level 2 read then occurs, the system will stay in a C2 state until a break event occurs. A Level 2 read has higher priority than the Thermal Override. However, if THRM# is still active after a break event occurs, the system will go back to the C0 state and STPCLK# will throttle (due to the THRM# signal).
- After an exit from a C2 state (due to a Break event), and if the THTL\_EN bit is still set, or if a Thermal Override is still occurring, the system will continue to throttle STPCLK#. Depending on the time of break event, the first transition on STPCLK# active can be delayed by up to one period.
- If in the C2 state after C1 (Halt), break event will return system to C1 state. If break event was an interrupt, then the INTR will be sent to the processor and system will transition from C1 to C0 state.

### 5.12.5 Sleep States

#### 5.12.5.1 Sleep State Overview

The ICH directly supports different sleep states (S1-S5), which are entered by setting the SLP\_EN bit, or due to a Power Button Override. The entry to the Sleep states are based on several assumptions:

- Entry to a Cx state is mutually exclusive with entry to a Sleep state. This is because the processor can only perform one register access at a time. A request to Sleep always has higher priority than throttling.
- Prior to setting the SLP\_EN bit, the software will turn off processor-controlled throttling. Note that thermal throttling cannot be disabled, but setting the SLP\_EN bit will disable thermal throttling (since S1-S5 sleep state has higher priority).
- Upon exit from the ICH-controlled Sleep states, the WAK\_STS bit will be set.

#### 5.12.5.2 Initiating Sleep State

Sleep states (S1-S5) are initiated by:

- Setting the desired type in the SLP\_TYP field and setting the SLP\_EN bit.
- Pressing the PWRBTN# Signal for more than 4 seconds to cause a Power Button Override event.

In case of the PWRBTN# override event, the transition to the S5 will be less graceful, since there will be no dependencies on observing Stop-Grant cycles from the processor or on clocks other than the RTC clock.

Depending on the type of Sleep state desired, the next step the hardware takes will be:

**Table 5-37. Sleep Types**

| Sleep Type | Comment  |
|------------|--|
| S1         | ICH asserts the processor SLP# signal. This will lower the processor's power consumption. No snooping is possible in this state.   |
| S2         | Not supported.   |
| S3         | ICH asserts SLP_S3#. The SLP_S3# signal will control the power to non-critical circuits. Power will only be retained to devices needed to wake from this sleeping state, as well as to the memory. |
| S4         | ICH asserts SLP_S3# and SLP_S5#. The SLP_S5# will shut off the power to the memory subsystem. Only devices needed to wake from this state should be powered.                                       |
| S5         | Same as S4. ICH asserts SLP_S3# and SLP_S5#. The SLP_S5# signal will shut off the power to the memory subsystem. Only devices needed to wake from this state should be powered.                    |

### 5.12.5.3 Exiting Sleep States

Sleep states (S1-S5) are exited based on Wake events. The Wake events will force the system to a full on state (S0), although some non-critical subsystems will still be shut off and have to be brought back manually. For example, the hard disk may be shut off during a sleep state, and have to be enabled via a GPIO pin before it can be used.

To enable Wake Events, the possible causes of wake events (and their restrictions) are shown in Table 5-38.

**Table 5-38. Causes of Wake Events**

| Cause        | States Can Wake From | How Enabled                         |
|--------------|----------------------|-------------------------------------|
| RTC Alarm    | S1 - S5<br>(Note 2)  | Set RTC_EN bit in PM1_EN Register   |
| Power Button | S1 - S5<br>(Note 1)  | Always enabled as Wake event        |
| GPIO[0:n]    | S1 - S5<br>(Note 2)  | GPE1_EN register                    |
| USB          | S1 - S4              | Set USB_EN bit in GPE0_EN Register  |
| RI#          | S1 - S5<br>(Note 2)  | Set RI_EN bit in GPE0_EN Register   |
| AC97         | S1 - S4              | Set AC97_EN bit in GPE0_EN Register |
| PME#         | S1 - S5<br>(Note 2)  | Set PME_EN bit in GPE0_EN Register. |
| SMBALERT#    | S1-S4                | SMB_WAK_EN in the GPE0 Register     |

**NOTES:**

1. If in the S5 state due to a powerbutton override, the only wake event is power button.
2. This will be a wake event from S5 only if the sleep state was entered by setting the SLP\_EN and SLP\_TYP bits via software.

It is important to understand that the various GPIs have different levels of functionality when used as wake events. The GPIs that reside in the core power well can only generate wake events from an S1 state. Also only certain GPIs are “ACPI Compliant,” meaning that their Status and Enable bits reside in ACPI I/O space. Table 5-39 summarizes the use of GPIs as wake events.

**Table 5-39. GPI Wake Events**

| GPI                | Power Well | Wake From | Notes          |
|--------------------|------------|-----------|----------------|
| GPI[1:0], GPI[7:5] | Core       | S1        |                |
| GPI[13:8]          | Resume     | S1-S5     | ACPI Compliant |
| GPIO[28:27]        | Resume     | S1-S5     |                |

The latency to exit the various Sleep states varies greatly and is heavily dependent on power supply design. Approximations are shown in Table 5-40. The time indicates from when the Wake event occurs (signal transition) to when the processor is allowed to start its first cycle (CPURST# goes inactive). There will be very large additional delays for the processor to execute sufficient amounts of BIOS to invoke the OS (such as coming out of S1-S3) or spinning up the hard drive (such as coming out of S4 or S5).

**Table 5-40. Sleep State Exit Latencies**

| State | Latency  |
|-------|--|
| S1    | <1 ms. Based on wake event to STPCLK# high + re-enumeration of PCI bus, USB, CardBus, etc. Must also add PLL spin-up times |
| S3    | Power Supply ramp + 20 ms  |
| S4    | Power Supply ramp + 20 ms  |
| S5    | Power Supply ramp + 20 ms  |

#### 5.12.5.4 Sx-G3-Sx, Handling Power Failures

Power failures can occur if the AC power is cut (a real power failure) or if the system is unplugged. Depending on when the power failure occurs and how the system is designed, different transitions could occur due to a power failure.

The AFTER\_G3 bit provides the ability to program whether or not the system should boot once power returns after a power loss event. If the policy is to not boot, the system will remain in an S5 state (unless previously in S4). There are only three possible events that will wake the system after a power failure.

1. PWRBTN#: PWRBTN# is always enabled as a wake event. When RSMRST# is low (G3 state), the PWRBTN\_STS bit is reset. When the ICH exits G3 after power returns (RSMRST# goes high), the PWRBTN# signal is already high (because Vcc-standby goes high before RSMRST# goes high) and the PWRBTN\_STS bit is 0.
2. RI#: RI# does not have an internal pull-up. Therefore, if this signal is enabled as a wake event, it is important to keep this signal powered during the power loss event. If this signal goes low (active), when power returns the RI\_STS bit will be set and the system will interpret that as a wake event.
3. RTC Alarm. The RTC\_EN bit is in the RTC well and is preserved after a power loss. Like PWRBTN\_STS the RTC\_STS bit is cleared when RSMRST# goes low.

The ICH monitors both PWROK and RSMRST# to detect for power failures. If PWROK goes low, the PWROK\_FLR bit is set. If RSMRST# goes low, PWR\_FLR is set.

**Note:** Although PME\_EN is in the RTC well, this signal cannot wake the system after a power loss. PME\_EN and PME\_STS bits are cleared by RSMRST#

**Table 5-41. Transitions Due To Power Failure**

| State at Power Failure | AFTERG0_EN bit | Transition When Power Returns |
|------------------------|----------------|-------------------------------|
| S0                     | 1              | S5                            |
|                        | 0              | S0                            |
| S1                     | 1              | S5                            |
|                        | 0              | S0                            |
| S3                     | 1              | S5                            |
|                        | 0              | S0                            |
| S4                     | 1              | S4                            |
|                        | 0              | S0                            |
| S5                     | 1              | S5                            |
|                        | 0              | S0                            |

## 5.12.6 Thermal Management

The ICH has mechanisms to assist with managing thermal problems in the system.

### 5.12.6.1 THRM# Signal

The THRM# signal is used as a status input for a thermal sensor. Based on the THRM# signal going active, the ICH generates an SMI# or SCI (depending on SCI\_EN).

If the THRM\_POL bit is cleared to “0”, when the THRM# signal goes low, the THRM\_STS bit will be set to “1”. This is an indicator that the thermal threshold has been exceeded. If the THRM\_EN bit is set, then when THRM\_STS goes active, either an SMI# or SCI is generated (depending on the SCI\_EN bit being set).

The power management software (BIOS or ACPI) can then take measures to start reducing the temperature. Examples include shutting off unwanted subsystems, or halting the processor.

By setting the THRM\_POL bit to high, another SMI# or SCI can optionally be generated when the THRM# signal goes back low. This allows the software (BIOS or ACPI) to turn off the cooling methods.

### 5.12.6.2 THRM# Initiated Passive Cooling

If the THRM# signal remains active for some time greater than 2 seconds and the ICH is in the S0/G0/C0 state, then the ICH enters an auto-throttling mode, in which it provides a duty cycle on the STPCLK# signal. This will reduce the overall power consumption by the system, and should cool the system. The intended result of the cooling is that the THRM# signal should go back inactive.

For all programmed values (001 - 111), THRM# going active will result in STPCLK# active for a minimum time of 12.5% and a maximum of 87.5%. The period is 1024 PCI clocks. Thus, the STPCLK# signal can be active for as little as 128 PCI clocks or as much as 896 PCI clocks. The

actual slowdown (and cooling) of the processor will depend on the instruction stream, because the processor is allowed to finish the current instruction. Furthermore, the ICH waits for the STOP-GRANT cycle before starting the count of the time the STPCLK# signal is active.

When THRM# goes inactive, the throttling will stop. However, there is a small window where the ICH may assert STPCLK# for one more throttling period after THERM# goes inactive.

In case that the ICH is already attempting throttling because the THTL\_EN bit is set, the duty cycle associated with the THRM# signal will have higher priority.

If the ICH is in the C2, or S1-S5 states, then no throttling will be caused by the THRM# signal being active.

### 5.12.6.3 Processor Initiated Passive Cooling (Via Programmed Duty Cycle on STPCLK#)

Using the THTL\_EN and THTL\_DTY bits, the ICH can force a programmed duty cycle on the STPCLK# signal. This will reduce the effective instruction rate of the processor and cut its power consumption and heat generation.

### 5.12.6.4 Active Cooling

Active cooling involves fans. The GPIO signals from the ICH can be used to turn on/off a fan.

## 5.12.7 Event Input Signals and Their Usage

The ICH has various input signals that trigger specific events. This section describes those signals and how they should be used.

### 5.12.7.1 PWRBTN#—Power Button

The ICH PWRBTN# signal operates as a “Fixed Power Button” as described in the ACPI specification. PWRBTN# signal has a 16 ms de-bounce on the input. The state transition descriptions are included in Table 5-42. N

**Note:** The transitions start as soon as the PWRBTN# is pressed (but after the debounce logic), and does not depend on when the Power Button is released.

Table 5-42. Transitions Due to Power Button

| Present State | Event   | Transition/Action                           | Comment  |
|---------------|---|---|--|
| S0/Cx         | PWRBTN# goes low                                    | SMI# or SCI generated (depending on SCI_EN) | Software will typically initiate a Sleep state.                                |
| S1-S5         | PWRBTN# goes low                                    | Wake Event. Transitions to S0 state.        | Standard wakeup  |
| G3            | PWRBTN# pressed                                     | None  | No effect since no power. Not latched nor detected.                            |
| S0-S4         | PWRBTN# held low for at least 4 consecutive seconds | Unconditional transition to S5 state.       | No dependence on processor (such as Stop-Grant cycles) or any other subsystem. |

**Power Button Override:** If PWRBTN# is observed active for at least 4 consecutive seconds, then the state machine should unconditionally transition to the G2/S5 state, regardless of present state (S0-S4). In this case, the transition to the G2/S5 state should not depend on any particular response from the processor (such as a Stop-Grant cycle), nor any similar dependency from any other subsystem.

The PWRBTN# status is readable to check if the button is currently being pressed or has been released. The status is taken after the de-bounce, and is readable via the PWRBTN\_LVL bit.

**Note:** The 4-second PWRBTN# assertion should only be used if a system lock-up has occurred. The 4-second timer starts counting when the ICH is in a S0 state. If the PWRBTN# signal is asserted and held active when the system is in a suspend state (S1-S5), the assertion will cause a wake event. Once the system has resumed to the S0 state, the 4-second timer will start.

**Sleep Button:** The ACPI specification defines an optional Sleep button. It differs from the power button in that it only is a request to go from S0 to S1-S4 (not S5). Also, in an S5 state, the Power Button can wake the system, but the Sleep Button cannot.

Although the ICH does not include a specific signal designated as a Sleep Button, one of the GPIO signals can be used to create a “Control Method” Sleep Button. This requires some AML code to be developed. Example code is provided in the ACPI specification.

### 5.12.7.2 RI#—Ring Indicate

The Ring Indicator can cause a wake event (if enabled) from the S1-S5 states. Table 5-43 shows when the wake event is generated or ignored in different states. If in the G0/S0/Cx states, the UART will configured to cause an interrupt based on RI# active, and the interrupt will be set up as a Break event.

Table 5-43. Transitions Due to RI# signal

| Present State | Event      | RI_EN | Event      |
|---------------|------------|-------|------------|
| S0            | RI# Active | X     | Ignored    |
| S1-S5         | RI# Active | 0     | Ignored    |
|               |            | 1     | Wake Event |

Filtering/Debounce on RI# will not be done in ICH. Can be in modem or external.

### 5.12.7.3 PME#—PCI Power Management Event

The PME# signal comes from a PCI device to request that the system be restarted. The PME# signal can generate an SMI#, SCI, or optionally a Wake event. The event occurs when the PME# signal goes from high to low. No event is caused when it goes from low to high.

### 5.12.8 Ait Access Mode

Before entering a low power state, several registers from powered down parts may need to be saved. In the majority of cases, this is not an issue, as registers have read and write paths. However, several of the ISA compatible registers are either read only or write only. To get data out of write-only registers, and to restore data into read-only registers, the ICH implements an alternate access mode.



### 5.12.8.1 Write Only Registers with Read Paths in Alternate Access Mode

The registers described in Table 5-44 have read paths in alternate access mode. The access number field in the table indicates which register will be returned per access to that port.

Table 5-44. Write Only Registers with Read Paths in Alternate Access Mode (Sheet 1 of 2)

| I/O  | # of | Restore Data |                                   | I/O  | # of | Restore Data |  |
|------|------|--------------|-----------------------------------|------|------|--------------|--|
| Addr | Rds  | Access       | Data                              | Addr | Rds  | Access       | Data   |
| 00h  | 2    | 1            | DMA Chan 0 base address low byte  | 40h  | 7    | 1            | Timer Counter 0 status, bits [5:0]           |
|      |      | 2            | DMA Chan 0 base address high byte |      |      | 2            | Timer Counter 0 base count low byte          |
| 01h  | 2    | 1            | DMA Chan 0 base count low byte    |      |      | 3            | Timer Counter 0 base count high byte         |
|      |      | 2            | DMA Chan 0 base count high byte   |      |      | 4            | Timer Counter 1 base count low byte          |
| 02h  | 2    | 1            | DMA Chan 1 base address low byte  |      |      | 5            | Timer Counter 1 base count high byte         |
|      |      | 2            | DMA Chan 1 base address high byte |      |      | 6            | Timer Counter 2 base count low byte          |
| 03h  | 2    | 1            | DMA Chan 1 base count low byte    |      |      | 7            | Timer Counter 2 base count high byte         |
|      |      | 2            | DMA Chan 1 base count high byte   | 41h  | 1    |              | Timer Counter 1 status, bits [5:0]           |
| 04h  | 2    | 1            | DMA Chan 2 base address low byte  | 42h  | 1    |              | Timer Counter 2 status, bits [5:0]           |
|      |      | 2            | DMA Chan 2 base address high byte | 70h  | 1    |              | Bit 7 = NMI Enable, Bits [6:0] = RTC Address |
| 05h  | 2    | 1            | DMA Chan 2 base count low byte    | C4h  | 2    | 1            | DMA Chan 5 base address low byte             |
|      |      | 2            | DMA Chan 2 base count high byte   |      |      | 2            | DMA Chan 5 base address high byte            |
| 06h  | 2    | 1            | DMA Chan 3 base address low byte  | C6h  | 2    | 1            | DMA Chan 5 base count low byte               |
|      |      | 2            | DMA Chan 3 base address high byte |      |      | 2            | DMA Chan 5 base count high byte              |
| 07h  | 2    | 1            | DMA Chan 3 base count low byte    | C8h  | 2    | 1            | DMA Chan 6 base address low byte             |
|      |      | 2            | DMA Chan 3 base count high byte   |      |      | 2            | DMA Chan 6 base address high byte            |
| 08h  | 6    | 1            | DMA Chan 0-3 Command <sup>2</sup> | CAh  | 2    | 1            | DMA Chan 6 base count low byte               |
|      |      | 2            | DMA Chan 0-3 Request              |      |      | 2            | DMA Chan 6 base count high byte              |
|      |      | 3            | DMA Chan 0 Mode: Bits(1:0) = "00" | CCh  | 2    | 1            | DMA Chan 7 base address low byte             |
|      |      | 4            | DMA Chan 1 Mode: Bits(1:0) = "01" |      |      | 2            | DMA Chan 7 base address high byte            |
|      |      | 5            | DMA Chan 2 Mode: Bits(1:0) = "10" | CEh  | 2    | 1            | DMA Chan 7 base count low byte               |
|      |      | 6            | DMA Chan 3 Mode: Bits(1:0) = "11" |      |      | 2            | DMA Chan 7 base count high byte              |
| 20h  | 12   | 1            | PIC ICW2 of Master controller     | D0h  | 6    | 1            | DMA Chan 4-7 Command <sup>2</sup>            |

Table 5-44. Write Only Registers with Read Paths in Alternate Access Mode (Sheet 2 of 2)

| I/O  | # of | Restore Data |  | I/O  | # of | Restore Data |                                    |
|------|------|--------------|--|------|------|--------------|------------------------------------|
| Addr | Rds  | Access       | Data                                       | Addr | Rds  | Access       | Data                               |
|      |      | 2            | PIC ICW3 of Master controller              |      |      | 2            | DMA Chan 4-7 Request               |
|      |      | 3            | PIC ICW4 of Master controller              |      |      | 3            | DMA Chan 4 Mode: Bits(1:0) = "00"  |
|      |      | 4            | PIC OCW1 of Master controller <sup>1</sup> |      |      | 4            | DMA Chan 5 Mode: Bits(1:0) = "01"  |
|      |      | 5            | PIC OCW2 of Master controller              |      |      | 5            | DMA Chan 6 Mode: Bits(1:0) = "10"  |
|      |      | 6            | PIC OCW3 of Master controller              |      |      | 6            | DMA Chan 7 Mode: Bits(1:0) = "11". |
|      |      | 7            | PIC ICW2 of Slave controller               |      |      |              |                                    |
|      |      | 8            | PIC ICW3 of Slave controller               |      |      |              |                                    |
|      |      | 9            | PIC ICW4 of Slave controller               |      |      |              |                                    |
|      |      | 10           | PIC OCW1 of Slave controller <sup>1</sup>  |      |      |              |                                    |
|      |      | 11           | PIC OCW2 of Slave controller               |      |      |              |                                    |
|      |      | 12           | PIC OCW3 of Slave controller               |      |      |              |                                    |

**NOTE:**

1. The OCW1 register must be read before entering Alternate Access Mode.
2. Bits 5, 3, 1, and 0 return 0.

### 5.12.8.2 PIC Reserved Bits

Many bits within the PIC are reserved, and must have certain values written in order for the PIC to operate properly. Therefore, there is no need to return these values in alternate access mode. When reading PIC registers from 20h and A0h, the reserved bits shall return the values listed in Table 5-45.

Table 5-45. PIC Reserved Bits Return Values

| PIC Reserved Bits | Value Returned |
|-------------------|----------------|
| ICW2(2:0)         | 000            |
| ICW4(7:5)         | 000            |
| ICW4(3:2)         | 00             |
| ICW4(0)           | 0              |
| OCW2(4:3)         | 00             |
| OCW3(7)           | 0              |
| OCW3(5)           | Reflects bit 6 |
| OCW3(4:3)         | 01             |

### 5.12.8.3 Read Only Registers with Write Paths in Alternate Access Mode

The registers described in Table 5-46 have write paths to them in alternate access mode. Software will restore these values after returning from a powered down state. These registers must be handled special by software. When in normal mode, writing to the base address/count register also writes to the current address/count register. Therefore, the base address/count must be written first, then the part is put into alternate access mode and the current address/count register is written.

**Table 5-46. Register Write Accesses in Alternate Access Mode**

| I/O Address | Register Write Value                   |
|-------------|--|
| 08h         | DMA Status Register for channel's 0-3. |
| D0h         | DMA Status Register for channel's 4-7. |

## 5.12.9 System Power Supplies, Planes, and Signals

### Power Plane Control with SLP\_S3# and SLP\_S5#

The SLP\_S3# output signal can be used to cut power to the system core supply, since it will only go active for the STR state (typically mapped to ACPI S3). Power must be maintained to the ICH Resume Well, and to any other circuits that need to generate Wake signals from the STR state.

Cutting power to the core may be done via the power supply, or by external FETs to the motherboard. The SLP\_S5# output signal can be used to cut power to the system core supply, as well as power to the system memory, since the context of the system is saved on the disk. Cutting power to the memory may be done via the power supply, or by external FETs to the motherboard.

### PWROK Signal

The PWROK input should go active based on the core supply voltages becoming valid. PWROK should go high at least 16 ms after the power is guaranteed valid.

**Note:** Traditional designs have a reset button logically AND'd with the PWROK signal from the power supply and the processor's voltage regulator module. If this is done with the ICH, the PWROK\_FLR bit will be set. The ICH treats this internally as if the RSMRST# signal had gone active. However, it is not treated as a full power failure. If PWROK goes inactive and then active (but RSMRST# stays high), then the ICH will reboot (regardless of the state of the AFTERG3 bit). If the RSMRST# signal also goes low before PWROK goes high, then this is a full power failure, and the reboot policy is controlled by the AFTERG3 bit.

## Controlling Leakage and Power Consumption During Low-Power States

To control leakage in the system, various signals will tri-state or go low during some low-power states.

General principles

- All signals going to powered down planes (either internally or externally) must be either tri-stated or driven low.
- Signals with pull-up resistors should not be low during low-power states. This is to avoid the power consumed in the pull-up resistor.
- Buses should be halted (and held) in a known state to avoid a floating input. (perhaps to some other device) Floating inputs can cause extra power consumption.

Based on the above principles, the following measures are taken:

- During C2 or S3 state, the processor signals that have pull-ups will be tri-stated.
- During S3, all signals attached to powered down planes will be tri-stated or driven low.

### 5.12.10 Clock Generators

The clock generator is expected to provide the frequencies shown in Table 5-47.

Table 5-47. ICH Input Clocks

| Clock Name | Frequency    | Event  |
|------------|--------------|--|
| CLK48      | 48.00000 MHz | This should be running in C0, C1, C2. Will stop based on SLP_S3# being active.   |
| CLK14      | 14.31818 MHz | This should be running in C0, C1, C2. Will stop based on SLP_S3# being active.   |
| CLK66      | 66.66667 MHz | This should be running in C0, C1, and C2. Will stop based on SLP_S3# being active.   |
| APIC_CLK   | 16.67 MHz    | This should be running in C0, C1, and C2. Will stop based on SLP_S3# being active.   |
| PCICLK     | 33.33333 MHz | The PCI clock will start/ stop based on the S3, S4, or S5 states. The PCI clock is assumed always to be running in S0-S1 states. |

### 5.12.11 Legacy Power Management Theory of Operation

Instead of relying on ACPI software, legacy power management uses BIOS and various hardware mechanisms. The ICH has a greatly simplified method for legacy power management compared with previous generations.

The scheme relies on the concept of detecting when individual subsystems are idle, detecting when the whole system is idle, and detecting accesses are attempted to idle subsystems.

However, the OS is assumed to be at least APM enabled. Without APM calls, there is no quick way to know when the system is idle between keystrokes. The ICH does not support the burst modes found in previous components.

### APM Power Management

The ICH has a timer that, when enabled by the `1MIN_EN` bit in the SMI Control and Enable register, will generate an SMI# once per minute. The SMI handler can check for system activity by reading the `DEVACT_STS` register. If none of the system bits are set, the SMI handler can increment a software counter. When the counter reaches a sufficient number of consecutive minutes with no activity, the SMI handler can then put the system into a lower power state.

If there is activity, various bits in the `DEVACT_STS` register will be set. Software clears the bits by writing a 1 to the bit position.

The `DEVACT_STS` register allows for monitoring various internal devices, or Super I/O devices (SP, PP, FDC) on LPC or PCI, keyboard controller accesses, or audio functions on LPC or PCI. Other PCI activity can be monitored by checking the PCI interrupts.

## 5.13 System Management (D31:F0)

### 5.13.1 Overview of System Management Functions

The ICH provides various functions to make a system easier to manage and to lower the Total Cost of Ownership (TCO) of the system. Features and functions can be augmented via external A/D converters and GPIO, as well as an external microcontroller.

The following features and functions are supported by the ICH:

- Processor present detection.
  - Detects if the processor fails to fetch the first instruction after reset.
- Various Error detection, reported by the host controller
  - Can generate SMI#, SCI, SERR, NMI, or TCO interrupt
- Intruder Detect input
  - Can generate TCO interrupt or SMI# when the system cover is removed.
- Alert On LAN (ICH: 82801AA only)
  - Sends hardcoded SMBus message over ALERTCLK and ALERTDATA pins to LAN controller.

**Note:** Voltage ID from the processor can be read via GPI signals.

### 5.13.2 Theory of Operation

The System Management functions are designed to allow the system to diagnose failing subsystems. The intent of this logic is that some of the system management functionality be provided without the aid of an external microcontroller.

#### Detecting a System Lockup

When the processor is reset, it is expected to fetch its first instruction. If the processor fails to fetch the first instruction after reset, the TCO timer will timeout twice and the ICH will assert `PCIRST#`.

### Handling an Intruder

The ICH has an input signal, INTRUDER#, that can be attached to a switch that is activated by the system's case being open. This input has a 2 RTC clock debounce. If INTRUDER# goes active (after the debouncer), this will set the INTRD\_DET bit in the TCO\_STS register. The INTRD\_SEL bits in the TCO\_CNT register can enable the ICH to cause an SMI# or interrupt. The BIOS or interrupt handler can then cause a transition to the S5 state by writing to the SLP\_EN bit.

The software can also directly read the status of the INTRUDER# signal (high or low) by clearing and then reading the INTRD\_DET bit. This allows the signal to be used as a GPI if the intruder function is not required.

#### 5.13.2.1 Handling an ECC Error or Other Memory Error

The host controller provides a message to indicate that it would like to cause an SMI#, SCI, SERR#, or NMI. The software must check the host controller as to the exact cause of the error.

#### 5.13.3 Alert on LAN\* (ICH: 82801AA only)

ICH can interface directly with an Alert On LAN enabled LAN controller to report messages to a network management console without the aid of the system processor. This is crucial in cases where the processor is malfunctioning or cannot function due to being in a low-power state.

The basic scheme is for the ICH to send specific messages via the SMBus Interface to the LAN controller. Upon receiving the SMBus message, the network controller has a prepared Ethernet message that it can send to a network management console. The prepared message is typically stored in a non-volatile memory connected directly to the network controller.

Messages will be sent by the ICH over the SMBus either because a specific event has occurred, or they will be sent periodically (also known as a heartbeat). The event and heartbeat messages will have the exact same format. The event messages will be sent based on events occurring. The heartbeat messages will be sent every 30 to 32 seconds. Whenever an event occurs the ICH will send a new message and increment the SEQ[3:0] field. For heartbeat messages the sequence number will not increment.

If the policy is for the ICH to reboot the system after a hardware lockup, then the ICH will not send an Alert On LAN message. Instead, the BIOS will detect the lockup and send the appropriate message.

If the policy is for the ICH not to reboot after a hardware lockup, then the ICH will send an Alert On LAN message with the Watchdog (WD) Event Status bit set. This message will be sent as soon as the lockup is detected. The message will be sent with the next incremented sequence number. Until the locked system is suspended (via a power button override) the ICH will continue to send Alert On LAN messages every heartbeat period (30-32 seconds). If another event occurs prior to a power button override, the ICH will send another Alert On LAN message with the next incremented sequence number and appropriate status bit set.

If a boot is unsuccessful (processor does not fetch the first instruction), then the ICH will send an Alert On LAN message with the processor event status bit set and the next incremented sequence number. This message will be sent as soon as the lockup is detected (2 TCO timer timeouts).

Since some SMBus devices will not be powered during some low power states, a separate SMBus interface is provided to connect the ICH to the Alert On LAN network controller. The signals that comprise this interface (referred to as ALERTCLK and ALERTDATA) are multiplexed with GPIO[27:28]. At reset, these signals default to ALERTCLK and ALERTDATA. Note that Alert on LAN messages will be broadcast *only* over this separate interface, and the ICH will look for an ACK from the LAN controller on the ALERTDATA signal.

If a unified SMBus architecture is desired, the SMBDATA and ALERTDATA (and SMBCLK and ALERTCLK) signals can be “ORed” together since they are both open drain driven signals. However, the ICH does not check to see if the SMBus is busy before broadcasting an Alert on LAN message using the ALERTCLK and ALERTDATA signals. So if the two interfaces are tied together it is important that there be no SMBus masters other than the ICH in the system. It will also be necessary to provide any necessary logic to isolate devices powered by standby from devices powered by core power.

If the system is in a G1 (S1-S4) state the ICH will send a heartbeat message every 30-32 seconds. If an event occurs prior to the system being shutdown, the ICH will immediately send an event message with the next incremented sequence number. After the event message the ICH will resume sending heartbeat messages.

Note that for all message fields more than 1 bit, the most significant bit (MSB) is transferred first. The most significant bit is on the left. For example, for 100000 the “1” is the MSB.

All Alert On LAN messages are broadcast over the ALERTCLK and ALERTDATA pins which are multiplexed with GPIO27 and GPIO28. There are two types of Alert On LAN messages; heartbeat messages and event messages. Heartbeat messages occur every 32 seconds. When coming out of G3, the first heartbeat message could occur anytime within the first 32 seconds. The second heartbeat message will occur 32 seconds after the first. However, event messages occur right after certain events occur. One such event is the act of entering into the pre-boot state.

Coming out of a G3 state ALERTCLK/GPIO27 and ALERTDATA/GPIO28 default to ALERTCLK and ALERTDATA. These signals can be programmed as GPIOs. However, before the BIOS has a chance to program these signals to GPIOs, the pre-boot event message and possibly a heartbeat message could be driven onto ALERTCLK and ALERTDATA.

If ALERTCLK and ALERTDATA are left floating, the SMBus state machine could get stuck in a busy state indefinitely when an Alert On LAN message goes out. To prevent this condition, the ICH requires external pull-ups to 3.3V standby on ALERTCLK and ALERTDATA.

If ALERTCLK and ALERTDATA are to be used as GPIOs, they need to be pulled up to 3.3V standby to avoid the SMBus from getting stuck. Also, when coming out of a G3 state, the devices that are connected to the GPIOs must be able to withstand the ICH driving Alert On LAN messages on these pins until the BIOS has a chance to program these pins as GPIOs. Examples of devices that would not be affected by this behavior include LEDs.

Table 5-48. Alert On LAN Message (ICH: 82801AA only)

| Field                          | Comment  |
|--------------------------------|--|
| Cover Tamper Status            | 1 = This bit will be set if the intruder detect bit is set (INTRD_DET).  |
| Temp Event Status              | 1 = This bit will be set if the ICH THERM# input signal is asserted.   |
| Processor Missing Event Status | <p>1 = This bit will be set if the processor failed to fetch the first instruction.</p> <p>If the NO-REBOOT bit (D31:F0, Offset D4h, bit 1) is set and the SECOND_TO_STS bit (TCO I/O offset 06h, bit 1) is set, and the BOOT_STS bit (TCO I/O offset 06h, bit 2) is set then the ICH will set the processor Missing Event Status bit in the Alert On LAN message.</p> <p>If the NO-REBOOT bit is not set, and the SECOND_TO_STS bit is set, the ICH will attempt to reboot. After the reboot, the SECOND_TO_STS bit will still be set. If the processor fails to fetch the first instruction, the BOOT_STS bit gets set. When the TCO timer times out again, the ICH will set the processor Missing Event Status bit in the Alert On LAN message.</p> |
| TCO Timer Event Status         | 1 = This bit is set when the TCO timer expires.  |
| Software Event Status          | 1 = This bit is set when software writes a 1 to the SEND_NOW bit.  |
| GPIO Status                    | <p>1 = This bit is set when GPIO[11] signal is low.</p> <p>0 = This bit is cleared when GPIO[11] signal is high.</p> <p>An event message is triggered on an transition of GPIO[11].</p>  |
| SEQ[3:0]                       | This is a sequence number. It will initially be 0, and will increment each time the ICH sends a new message. Upon reaching 1111, then the sequence number will roll over to 0000. MSB (SEQ3) sent first.   |
| System Power State             | 00 = G0, 01 = G1, 10 = G2, 11 = Pre-Boot. MSB sent first<br>See Table 5-49 for a definition for each of these states.  |
| MESSAGE1                       | Will be the same as the MESSAGE1 Register. MSB sent first.   |
| MESSAGE2                       | Will be the same as the MESSAGE2 Register. MSB sent first.   |
| WDSTATUS                       | Will be the same as the WDSTATUS Register. MSB sent first.   |



**Table 5-49. System Power State Definition (ICH: 82801AA only)**

| State    | Enter State  | Exit State   |
|----------|--|--|
| Pre-Boot | ICH has left G1 or G2 by deasserting SLP_S3#                         | CPURST# deasserted or powerbutton override                       |
| G0       | CPURST# deasserted   | Write to SLP_EN register or powerbutton override                 |
| G1       | Write to SLP_EN register with SLP_TYP for S1-S4                      | ICH deasserts SLP_S3# after a wake event or powerbutton override |
| G2       | Write to SLP_EN register with SLP_TYP for S5 or powerbutton override | ICH deasserts SLP_S3# after a wake event                         |

## 5.14 IDE Controller (D31:F1)

The IDE interface of the ICH can support several types of data transfers:

- **Programmed I/O (PIO):** Processor is in control of the data transfer.
- **8237 style DMA:** DMA protocol that resembles the DMA on the ISA bus, although it does not use the 8237 in the ICH. This protocol off loads the processor from moving data. This allows higher transfer rate of up to 16 MB/s.
- **Ultra ATA/33:** DMA protocol that redefines signals on the IDE cable to allow both host and target throttling of data and transfer rates of up to 33 MB/s.
  - Note:* The ICH0 (82801AB) does not support Ultra ATA/66. If software attempts to program an IDE drive to run ATA/66 timings, the ICH0 will see short strobes (from ATA/33 perspective) and will throttle the performance. This will result in transfer rates that are slower than if the drive were programmed for ATA/33 mode.
- **Ultra ATA/66 (ICH: 82801AA only):** DMA protocol that redefines signals on the IDE cable to allow both host and target throttling of data and transfer rates of up to 66 MB/s.

The ICH has integrated termination series resistors on the data and control lines as shown in Table 5-50.

**Table 5-50. IDE Integrated Series Termination Resistors**

| Signal   | Integrated Series Termination Resistor Value | Notes |
|--|--|-------|
| PDD[15:0], SDD[15:0]   | approximately 33 ohms                        | 1     |
| PDIOW#, SDIOW#, PDIOR#, PDIOW#, PDREQ, SDREQ, PDDACK#, SDDACK#, PIORDY, SIORDY, PDA[2:0], SDA[2:0], PDCS1#, SDCS1#, PDCS3#, SDCS3#, IRQ14, IRQ15 | approximately 22 ohms                        | 2     |

**NOTES:**

1. Simulation data indicates that these integrated series termination resistors are a nominal 33 ohms but can range from 28 ohms to 46 ohms.
2. Simulation data indicates that these integrated series termination resistors are a nominal 22 ohms but can range from 19 ohms to 29 ohms.

## 5.14.1 PIO Transfers

The ICH IDE controller includes both compatible and fast timing modes. The fast timing modes can be enabled only for the IDE data ports. All other transactions to the IDE registers are run in single transaction mode with compatible timings. The ICH IDE signals are controlled with the granularity of the PCI clock.

Up to 2 IDE devices may be attached per IDE connector (drive 0 and drive 1). The IDETIM and SIDETIM Registers permit different timing modes to be programmed for drive 0 and drive 1 of the same connector.

The Ultra ATA/33 synchronous DMA timing modes can also be applied to each drive by programming the SDMACTL and SDMATIM registers. When a drive is enabled for synchronous DMA mode operation, the DMA transfers are executed with the synchronous DMA timings. The PIO transfers are executed using compatible timings or fast timings if also enabled.

### 5.14.1.1 PIO IDE Timing Modes

IDE data port transaction latency consists of startup latency, cycle latency, and shutdown latency.

Startup latency is incurred when a PCI master cycle targeting the IDE data port is decoded and the DA[2:0] and CSxx# lines are not set up. Startup latency provides the setup time for the DA[2:0] and CSxx# lines prior to assertion of the read and write strobes (DIOR# and DIOW#).

Cycle latency consists of the I/O command strobe assertion length and recovery time. Recovery time is provided so that transactions may occur back-to-back on the IDE interface (without incurring startup and shutdown latency) without violating minimum cycle periods for the IDE interface. The command strobe assertion width for the enhanced timing mode is selected by the IDETIM Register and may be set to 2, 3, 4, or 5 PCI clocks. The recovery time is selected by the IDETIM Register and may be set to 1, 2, 3, or 4 PCI clocks.

If IORDY is asserted when the initial sample point is reached, no wait states are added to the command strobe assertion length. If IORDY is negated when the initial sample point is reached, additional wait states are added. Since the rising edge of IORDY must be synchronized, at least two additional PCI clocks are added.

Shutdown latency is incurred after outstanding scheduled IDE data port transactions (either a non-empty write post buffer or an outstanding read prefetch cycles) have completed and before other transactions can proceed. It provides hold time on the DA[2:0] and CSxx# lines with respect to the read and write strobes (DIOR# and DIOW#). Shutdown latency is 2 PCI clocks in duration. The IDE timings for various transaction types are shown in Table 5-51.

Note that bit 2 (16 bit I/O recovery enable) of the ISA I/O Recovery Timer Register does not add wait states to IDE data port read accesses when any of the fast timing modes are enabled.

**Table 5-51. IDE Transaction Timings (PCI Clocks)**

| IDE Transaction Type     | Startup Latency | IORDY Sample Point (ISP) | Recovery Time (RCT) | Shutdown Latency |
|--------------------------|-----------------|--------------------------|---------------------|------------------|
| Non-Data Port Compatible | 4               | 11                       | 22                  | 2                |
| Data Port Compatible     | 3               | 6                        | 14                  | 2                |
| Fast Timing Mode         | 2               | 2 - 5                    | 1 - 4               | 2                |

### 5.14.1.2 IORDY Masking

The IORDY signal can be ignored and assumed asserted at the first IORDY Sample Point (ISP) on a drive by drive basis via the IDETIM Register.

### 5.14.1.3 PIO 32 Bit IDE Data Port Accesses

A 32 bit PCI transaction run to the IDE data address (01F0h primary, 0170h secondary) results in two back to back 16 bit transactions to the IDE data port. The 32 bit data port feature is enabled for all timings, not just enhanced timing. For compatible timings, a shutdown and startup latency is incurred between the two 16-bit halves of the IDE transaction. This guarantees that the chip selects will be deasserted for at least 2 PCI clocks between the 2 cycles.

### 5.14.1.4 PIO IDE Data Port Prefetching and Posting

The ICH can be programmed via the IDETIM registers to allow data to be posted to and prefetched from the IDE data ports.

Data prefetching is initiated when a data port read occurs. The read prefetch eliminates latency to the IDE data ports and allows them to be performed back to back for the highest possible PIO data transfer rates. The first data port read of a sector is called the demand read. Subsequent data port reads from the sector are called prefetch reads. The demand read and all prefetch reads must be of the same size (16 or 32 bits).

Data posting is performed for writes to the IDE data ports. The transaction is completed on the PCI bus after the data is received by the ICH. The ICH will then run the IDE cycle to transfer the data to the drive. If the ICH write buffer is non-empty and an unrelated (non-data or opposite channel) IDE transaction occurs, that transaction will be stalled until all current data in the write buffer is transferred to the drive.

## 5.14.2 Bus Master Function

The ICH can act as a PCI Bus master on behalf of an IDE slave device. Two PCI Bus master channels are provided, one channel for each IDE connector (primary and secondary). By performing the IDE data transfer as a PCI Bus master, the ICH off-loads the processor and improves system performance in multitasking environments. Both devices attached to a connector can be programmed for bus master transfers, but only one device per connector can be active at a time.

### 5.14.2.1 Physical Region Descriptor Format

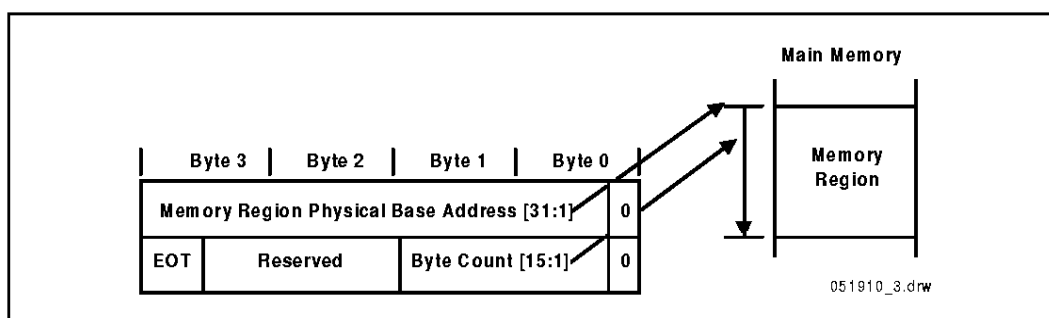
The physical memory region to be transferred is described by a Physical Region Descriptor (PRD). The PRDs are stored sequentially in a Descriptor Table in memory. The data transfer proceeds until all regions described by the PRDs in the table have been transferred. Note that the ICH bus master IDE function does not support memory regions or Descriptor tables located on ISA.

Descriptor Tables must be aligned on 64 Kbyte boundaries. Each PRD entry in the table is 8 bytes in length. The first 4 bytes specify the byte address of a physical memory region. This memory region must be DWORD aligned and must not cross a 64 KByte boundary. The next two bytes specify the size or transfer count of the region in bytes (64 KByte limit per region). A value of zero in these two bytes indicates 64 Kbytes (thus the minimum transfer count is 1). If bit 7 (EOT) of the last byte is a 1, it indicates that this is the final PRD in the Descriptor table. Bus master operation terminates when the last descriptor has been retired.

When the Bus Master IDE controller is reading data from the memory regions, bit 1 of the Base Address is masked and byte enables are asserted for all read transfers. The controller will read to a boundary of 64 bytes, regardless of byte count field of the PRD. Only the byte count value will be transferred to the drive however. When writing data, bit 1 of the Base Address is not masked and if set, will cause the lower WORD byte enables to be deasserted for the first DWORD transfer. The write to PCI will typically consist of a 32 byte cache line. If valid data ends prior to end of the cache line, the byte enables will be deasserted for invalid data.

The total sum of the byte counts in every PRD of the descriptor table must be equal to or greater than the size of the disk transfer request. If greater than the disk transfer request, the driver must terminate the bus master transaction (by setting bit 0 in the Bus Master IDE Command Register to 0) when the drive issues an interrupt to signal transfer completion.

Figure 5-12. Physical Region Descriptor Table Entry



### 5.14.2.2 Line Buffer

A single line buffer exists for the ICH Bus master IDE interface. This buffer is not shared with any other function. The buffer is maintained in either the read state or the write state. Memory writes are typically 4-Dword bursts and invalid DWords have C/BE[3:0]#=0Fh. The line buffer allows burst data transfers to proceed at peak transfer rates.

The Bus Master IDE Active bit in Bus Master IDE Status register is reset automatically when the controller has transferred all data associated with a Descriptor Table (as determined by EOT bit in last PRD). The IDE Interrupt Status bit is set when the IDE device generates an interrupt. These events may occur prior to line buffer emptying for memory writes. If either of these conditions exist, all PCI Master non-memory read accesses to ICH are retried until all data in the line buffers has been transferred to memory.

### 5.14.2.3 Bus Master IDE Timings

The timing modes used for Bus Master IDE transfers are identical to those for PIO transfers. The DMA Timing Enable Only bits in IDE Timing register can be used to program fast timing mode for DMA transactions only. This is useful for IDE devices whose DMA transfer timings are faster than its PIO transfer timings. The IDE device DMA request signal is sampled on the same PCI clock that DIOR# or DIOW# is deasserted. If inactive, the DMA Acknowledge signal is deasserted on the next PCI clock and no more transfers take place until DMA request is asserted again.

#### 5.14.2.4 Interrupts

The ICH is connected to IRQ14 for the primary interrupt and IRQ15 for the secondary interrupt. This connection is done from the ISA pin, before any mask registers. This implies the following:

- Bus Master IDE is operating under an interrupt based driver. Therefore, it will not operate under environments where the IDE device drives an interrupt but the interrupt is masked in the system.
- Bus Master IDE devices are connected directly off of ICH. IDE interrupts cannot be communicated through PCI devices or the serial stream.

#### 5.14.2.5 Bus Master IDE Operation

To initiate a bus master transfer between memory and an IDE device, the following steps are required:

1. Software prepares a PRD Table in system memory. The PRD Table must be dWord aligned and must not cross a 64K byte boundary.
2. Software provides the starting address of the PRD Table by loading the PRD Table Pointer Register. The direction of the data transfer is specified by setting the Read/Write Control bit. The interrupt bit and Error bit in the Status register are cleared.
3. Software issues the appropriate DMA transfer command to the disk device.
4. The bus master function is engaged by software writing a '1' to the Start bit in the Command Register. The first entry in the PRD table is fetched and loaded into two registers which are not visible by software, the Current Base and Current Count registers. These registers hold the current value of the address and byte count loaded from the PRD table. The value in these registers is only valid when there is an active command to an IDE device.
5. Once the PRD is loaded internally, the IDE device will receive a DMA acknowledge.
6. The controller transfers data to/from memory responding to DMA requests from the IDE device. The IDE device and the host controller may or may not throttle the transfer several times. When the last data transfer for a region has been completed on the IDE interface, the next descriptor is fetched from the table. The descriptor contents are loaded into the Current Base and Current Count registers.
7. At the end of the transfer the IDE device signals an interrupt.
8. In response to the interrupt, software resets the Start/Stop bit in the command register. It then reads the controller status followed by the drive status to determine if the transfer completed successfully.

The last PRD in a table has the End of List (EOL) bit set. The PCI bus master data transfers will terminate when the physical region described by the last PRD in the table has been completely transferred. The active bit in the Status Register will be reset and the DDRQ signal will be masked.

The 64 byte buffer is flushed (when in the write state) or invalidated (when in the read state) when a terminal count condition exists; that is, the current region descriptor has the EOL bit set and that region has been exhausted. The buffer is also flushed (write state) or invalidated (read state) when the Interrupt bit in the Bus Master IDE Status register is set. Software that reads the status register and finds the Error bit reset, and either the Active bit reset or the Interrupt bit set, can be assured that all data destined for system memory has been transferred and that data is valid in system memory. Table 5-52 describes how to interpret the Interrupt and Active bits in the Status Register after a DMA transfer has started.

Table 5-52. Interrupt/Active Bit Interaction Definition

| Interrupt | Active | Description   |
|-----------|--------|---|
| 0         | 1      | DMA transfer is in progress. No interrupt has been generated by the IDE device.   |
| 1         | 0      | The IDE device generated an interrupt. The controller exhausted the Physical Region Descriptors. This is the normal completion case where the size of the physical memory regions was equal to the IDE device transfer size.  |
| 1         | 1      | The IDE device generated an interrupt. The controller has not reached the end of the physical memory regions. This is a valid completion case where the size of the physical memory regions was larger than the IDE device transfer size.   |
| 0         | 0      | This bit combination signals an error condition. If the Error bit in the status register is set, then the controller has some problem transferring data to/from memory. Specifics of the error have to be determined using bus-specific information. If the Error bit is not set, then the PRD's specified a smaller size than the IDE transfer size. |

### 5.14.2.6 Error Conditions

IDE devices are sector based mass storage devices. The drivers handle errors on a sector basis; either a sector is transferred successfully or it is not. A sector is 512 bytes.

If the IDE device does not complete the transfer due to a hardware or software error, the command will eventually be stopped by the driver setting Command Start bit to zero when the driver times out the disk transaction. Information in the IDE device registers help isolate the cause of the problem.

If the controller encounters an error while doing the bus master transfers, it will stop the transfer (i.e., reset the Active bit in the Command register) and set the Error bit in the Bus Master IDE Status register. The controller does not generate an interrupt when this happens. The device driver can use device specific information (PCI Configuration Space Status register and IDE Drive Register) to determine what caused the error.

When a requested transfer does not complete properly, information in the IDE device registers (Sector Count) can be used to determine how much of the transfer was completed and to construct a new PRD table to complete the requested operation. In most cases the existing PRD table can be used to complete the operation.

### 5.14.2.7 8237 Like Protocol

The 8237 mode DMA is similar in form to DMA used on the ISA bus. This mode uses pins familiar to the ISA bus, namely a DMA Request, a DMA Acknowledge, and I/O read/write strobes. These pins have similar characteristics to their ISA counterparts in terms of when data is valid relative to strobe edges, and the polarity of the strobes; however, the ICH does not use the 8237 for this mode.

### 5.14.3 Ultra ATA/33 Protocol

Ultra ATA/33 is enabled through configuration register 48h in Device 31:Function 1 for each IDE device. The IDE signal protocols are significantly different under this mode than for the 8237 mode.

Ultra ATA/33 is a physical protocol used to transfer data between a Ultra ATA/33 capable IDE controller such as the ICH and one or more Ultra ATA/33 capable IDE devices. It utilizes the standard Bus Master IDE functionality and interface to initiate and control the transfer. Ultra ATA/33 utilizes a “source synchronous” signaling protocol to transfer data at rates up to 33 Mbytes/sec. The Ultra ATA/33 definition also incorporates a Cyclic Redundancy Checking (CRC-16) error checking protocol.

#### 5.14.3.1 Signal Descriptions

The Ultra ATA/33 protocol requires no extra signal pins on the IDE connector. It does redefine a number of the standard IDE control signals when in Ultra ATA/33 mode. These redefinitions are shown in Table 5-53. Read cycles are defined as transferring data from the IDE device to the ICH. Write cycles are defined as transferring data from ICH to IDE device.

**Table 5-53. Ultra ATA/33 Control Signal Redefinitions**

| Standard IDE Signal Definition | Ultra ATA/33 Read Cycle Definition | Ultra ATA/33 Write Cycle Definition | ICH Primary Channel Signal | ICH Secondary Channel Signal |
|--------------------------------|------------------------------------|-------------------------------------|----------------------------|------------------------------|
| DIOW#                          | STOP                               | STOP                                | PDIOW#                     | SDIOW#                       |
| DIOR#                          | DMARDY#                            | STROBE                              | PDIOR#                     | SDIOR#                       |
| IORDY                          | STROBE                             | DMARDY#                             | PIORDY                     | SIORDY                       |

The DIOW# signal is redefined as STOP for both read and write transfers. This is always driven by the ICH and is used to request that a transfer be stopped or as an acknowledgment to stop a request from the IDE device.

The DIOR# signal is redefined as DMARDY# for transferring data from the IDE device to the ICH (read). It is used by the ICH to signal when it is ready to transfer data and to add wait states to the current transaction. The DIOR# signal is redefined as STROBE for transferring data from the ICH to the IDE device (write). It is the data strobe signal driven by the ICH on which data is transferred during each rising and falling edge transition.

The IORDY signal is redefined as STROBE for transferring data from the IDE device to the ICH (read). It is the data strobe signal driven by the IDE device on which data is transferred during each rising and falling edge transition. The IORDY signal is redefined as DMARDY# for transferring data from the ICH to the IDE device (write). It is used by the IDE device to signal when it is ready to transfer data and to add wait states to the current transaction.

All other signals on the IDE connector retain their functional definitions during Ultra ATA/33 operation.

### 5.14.3.2 Operation

Initial setup programming consists of enabling and performing the proper configuration of ICH and the IDE device for Ultra ATA/33 operation. For the ICH, this consists of enabling Synchronous DMA mode and setting up appropriate Synchronous DMA timings.

When ready to transfer data to or from an IDE device, the Bus Master IDE programming model is followed. Once programmed, the drive and ICH control the transfer of data via the Ultra ATA/33 protocol. The actual data transfer consists of three phases, a start-up phase, a data transfer phase, and a burst termination phase.

The IDE device begins the start-up phase by asserting DMARQ signal. When ready to begin the transfer, the ICH will assert DMACK# signal. When DMACK# signal is asserted, the host controller will drive CS0# and CS1# inactive, DA0-DA2 low and the IDE device will drive IOCS16# inactive. For write cycles, the ICH will deassert STOP, wait for the IDE device to assert DMARDY#, and then drive the first data word and STROBE signal. For read cycles, the ICH will tri-state the DD lines, deassert STOP, and assert DMARDY#. The IDE device will then send the first data word and STROBE.

The data transfer phase continues the burst transfers with the data transmitter (ICH - writes, IDE device - reads) providing data and toggling STROBE. Data is transferred (latched by receiver) on each rising and falling edge of STROBE. The transmitter can pause the burst by holding STROBE high or low, resuming the burst by again toggling STROBE. The receiver can pause the burst by deasserting DMARDY# and resumes the transfers by asserting DMARDY#. The ICH will pause a burst transaction in order to prevent an internal line buffer over or under flow condition, resuming once the condition has cleared. It may also pause a transaction if the current PRD byte count has expired, resuming once it has fetched the next PRD.

The current burst can be terminated by either the transmitter or receiver. A burst termination consists of a Stop Request, Stop Acknowledge and transfer of CRC data. The ICH can stop a burst by asserting STOP, with the IDE device acknowledging by deasserting DMARQ. The IDE device stops a burst by deasserting DMARQ and the ICH acknowledges by asserting STOP. The transmitter then drives the STROBE signal to a high level. The ICH will then drive the CRC value onto the DD lines and deassert DMACK#. The IDE device will latch the CRC value on rising edge of DMACK#. The ICH will terminate a burst transfer if it needs to service the opposite IDE channel, if a Programmed I/O (PIO) cycle is executed to the IDE channel currently running the burst, or upon transferring the last data from the final PRD.

### 5.14.3.3 CRC Calculation

Cyclic Redundancy Checking (CRC-16) is used for error checking on Ultra ATA/33 transfers. The CRC value is calculated for all data by both the ICH and the IDE device over the duration of the Ultra ATA/33 burst transfer segment. This segment is defined as all data transferred with a valid STROBE edge from DDACK# assertion to DDACK# deassertion. At the end of the transfer burst segment, the ICH will drive the CRC value onto the DD[15:0] signals. It is then latched by the IDE device on deassertion of DDACK#. The IDE device compares the ICH CRC value to its own and reports an error if there is a mismatch.

### 5.14.3.4 Synchronous DMA Timings

The timings for Ultra ATA/33 are programmed into the Synchronous DMA Timing Register. The programmable timings include Cycle Time (CT) and Ready to Pause (RP) time. The Cycle Time represents the minimum pulse width of active data strobe (STROBE) signal. The Ready to Pause time represents the number of PCI clocks the ICH will wait from deassertion of DMARDY# to the assertion of STOP when it desires to stop a burst read transaction.



### 5.14.3.5 Ultra ATA/66 (ICH: 82801AA only)

The ICH (82801AA) supports both the Ultra ATA/33 and Ultra ATA/66 protocols. Ultra ATA/66 is similar to the Ultra ATA/33 scheme and is intended to be device driver compatible. The Ultra ATA/66 logic clocks at 66 MHz instead of 33 MHz, and can move 16-bits of data every 2 clocks (for a maximum of 66MBytes/sec).

To achieve the higher data rate, the timings are shortened and the quality of the cable is improved to reduce reflections, noise, and inductive coupling. Note that the improved cable is required and will still plug into the standard IDE connector.

The Ultra ATA/66 protocol also has a 44 Mbytes/sec mode where it uses three 66 MHz clocks. The Ultra ATA/66 protocol is enabled via config bits 3:0 at offset 54h.

During concurrent DMA or Ultra-DMA transfers, the ICH IDE interface will arbitrate between the primary and secondary IDE cables when a PRD (Physical Region Descriptor) entry expires.

## 5.15 USB Controller (Device 31:Function 2)

The ICH contains one USB Host Controller. The Host Controller includes the root hub with two separate USB ports. The ICH Host Controller supports the standard Universal Host Controller Interface (UHCI) Rev 1.0. Overcurrent detection on two USB ports is supported. The overcurrent signals can be used as GPIO if not needed. The ICH's USB controller is arbitrated as differently than standard PCI devices to improve arbitration latency.

### 5.15.1 Data Structures in Main memory

This section describes the details of the data structures used to communicate control, status, and data between software and the ICH: Frame Lists, Transfer Descriptors, and Queue Heads. Frame Lists are aligned on 4-Kbyte boundaries. Transfer Descriptors and Queue Heads are aligned on 16-byte boundaries.

#### 5.15.1.1 Frame List Pointer

The frame list pointer contains a link pointer to the first data object to be processed in the frame, as well as the control bits defined in Table 5-54.

Table 5-54. Frame List Pointer Bit Description

| Bit  | Description   |
|------|---|
| 31:4 | <b>Frame List Pointer (FLP).</b> This field contains the address of the first data object to be processed in the frame and corresponds to memory address signals [31:4], respectively.  |
| 3:2  | Reserved. These bits must be written as 0.  |
| 1    | <b>QH/TD Select (Q).</b> This bit indicates to the hardware whether the item referenced by the link pointer is a TD (Transfer Descriptor) or a QH (Queue Head). This allows the ICH to perform the proper type of processing on the item after it is fetched.<br>1 = QH<br>0 = TD |
| 0    | <b>Terminate (T).</b> This bit indicates to the ICH whether the schedule for this frame has valid entries in it.<br>1 = Empty Frame (pointer is invalid).<br>0 = Pointer is valid (points to a QH or TD).   |

### 5.15.1.2 Transfer Descriptor (TD)

Transfer Descriptors (TDs) express the characteristics of the transaction requested on USB by a client. TDs are always aligned on 16-byte boundaries, and the elements of the TD are shown in Figure 5-13. The 4 different USB transfer types are supported by a small number of control bits in the descriptor that the ICH interprets during operation. All Transfer Descriptors have the same basic, 32-byte structure. During operation, the ICH hardware performs consistency checks on some fields of the TD. If a consistency check fails, the ICH halts immediately and issues an interrupt to the system. This interrupt cannot be masked within the ICH.

Figure 5-13. Transfer Descriptor

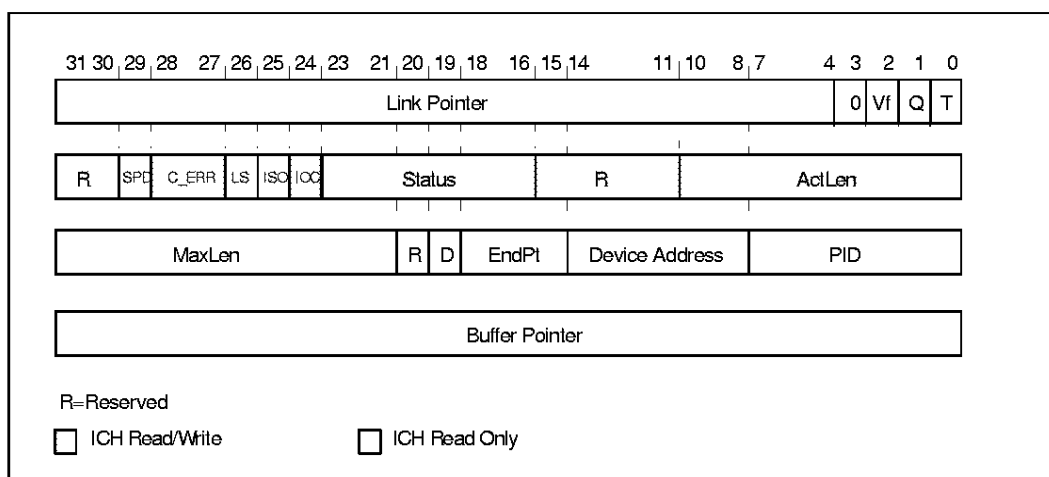


Table 5-55. TD Link Pointer

| Bit  | Description   |
|------|---|
| 31:4 | <b>Link Pointer (LP).</b> Bits [31:4] Correspond to memory address signals [31:4], respectively. This field points to another TD or QH.   |
| 3    | Reserved. Must be 0 when writing this field.  |
| 2    | <b>Depth/Breadth Select (VF).</b> This bit is only valid for queued TDs and indicates to the hardware whether it should process in a depth first or breadth first fashion. When set to depth first, it informs the ICH to process the next transaction in the queue rather than starting a new queue.<br>1 = Depth first.<br>0 = Breadth first.   |
| 1    | <b>QH/TD Select (Q).</b> This bit informs the ICH whether the item referenced by the link pointer is another TD or a QH. This allows the ICH to perform the proper type of processing on the item after it is fetched<br>1 = QH.<br>0 = TD.   |
| 0    | <b>Terminate (T).</b> This bit informs the ICH that the link pointer in this TD does not point to another valid entry. When encountered in a queue context, this bit indicates to the ICH that there are no more valid entries in the queue. A TD encountered outside of a queue context with the T bit set informs the ICH that this is the last TD in the frame.<br>1 = Link Pointer field not valid.<br>0 = Link Pointer field is valid. |

Table 5-56. TD Control and Status (Sheet 1 of 2)

| Bit             | Description  |                   |                   |    |                |    |         |    |          |    |          |       |                   |       |                   |           |     |                   |     |               |     |         |     |              |    |                 |     |                 |     |  |  |
|-----------------|--|-------------------|-------------------|----|----------------|----|---------|----|----------|----|----------|-------|-------------------|-------|-------------------|-----------|-----|-------------------|-----|---------------|-----|---------|-----|--------------|----|-----------------|-----|-----------------|-----|--|--|
| 31:30           | Reserved.  |                   |                   |    |                |    |         |    |          |    |          |       |                   |       |                   |           |     |                   |     |               |     |         |     |              |    |                 |     |                 |     |  |  |
| 29              | <p><b>Short Packet Detect (SPD).</b> When a packet has this bit set to 1 and the packet is an input packet, is in a queue; and successfully completes with an actual length less than the maximum length then the TD is marked inactive, the Queue Header is not updated and the USBINT status bit (Status Register) is set at the end of the frame. In addition, if the interrupt is enabled, the interrupt will be sent at the end of the frame.</p> <p>Note that any error (e.g., babble or FIFO error) prevents the short packet from being reported. The behavior is undefined when this bit is set with output packets or packets outside of queues.</p> <p>1 = Enable.<br/>0 = Disable.</p>   |                   |                   |    |                |    |         |    |          |    |          |       |                   |       |                   |           |     |                   |     |               |     |         |     |              |    |                 |     |                 |     |  |  |
| 28:27           | <p><b>Error Counter (C_ERR).</b> This field is a 2-bit down counter that keeps track of the number of Errors detected while executing this TD. If this field is programmed with a non zero value during setup, the ICH decrements the count and writes it back to the TD if the transaction fails. If the counter counts from one to zero, the ICH marks the TD inactive, sets the "STALLED" and error status bit for the error that caused the transition to zero in the TD. An interrupt will be generated to Host Controller Driver (HCD) if the decrement to zero was caused by Data Buffer error, Bit stuff error, or if enabled, a CRC or Timeout error. If HCD programs this field to zero during setup, the ICH will not count errors for this TD and there will be no limit on the retries of this TD.</p> <table border="0"> <thead> <tr> <th>Bits[28:27]</th> <th>Interrupt After</th> </tr> </thead> <tbody> <tr> <td>00</td> <td>No Error Limit</td> </tr> <tr> <td>01</td> <td>1 Error</td> </tr> <tr> <td>10</td> <td>2 Errors</td> </tr> <tr> <td>11</td> <td>3 Errors</td> </tr> </tbody> </table> <table border="0"> <thead> <tr> <th>Error</th> <th>Decrement Counter</th> <th>Error</th> <th>Decrement Counter</th> </tr> </thead> <tbody> <tr> <td>CRC Error</td> <td>Yes</td> <td>Data Buffer Error</td> <td>Yes</td> </tr> <tr> <td>Timeout Error</td> <td>Yes</td> <td>Stalled</td> <td>No*</td> </tr> <tr> <td>NAK Received</td> <td>No</td> <td>Bit stuff Error</td> <td>Yes</td> </tr> <tr> <td>Babble Detected</td> <td>No*</td> <td></td> <td></td> </tr> </tbody> </table> <p>* Detection of Babble or Stall automatically deactivates the TD. Thus, count is not decremented.</p> | Bits[28:27]       | Interrupt After   | 00 | No Error Limit | 01 | 1 Error | 10 | 2 Errors | 11 | 3 Errors | Error | Decrement Counter | Error | Decrement Counter | CRC Error | Yes | Data Buffer Error | Yes | Timeout Error | Yes | Stalled | No* | NAK Received | No | Bit stuff Error | Yes | Babble Detected | No* |  |  |
| Bits[28:27]     | Interrupt After  |                   |                   |    |                |    |         |    |          |    |          |       |                   |       |                   |           |     |                   |     |               |     |         |     |              |    |                 |     |                 |     |  |  |
| 00              | No Error Limit   |                   |                   |    |                |    |         |    |          |    |          |       |                   |       |                   |           |     |                   |     |               |     |         |     |              |    |                 |     |                 |     |  |  |
| 01              | 1 Error  |                   |                   |    |                |    |         |    |          |    |          |       |                   |       |                   |           |     |                   |     |               |     |         |     |              |    |                 |     |                 |     |  |  |
| 10              | 2 Errors   |                   |                   |    |                |    |         |    |          |    |          |       |                   |       |                   |           |     |                   |     |               |     |         |     |              |    |                 |     |                 |     |  |  |
| 11              | 3 Errors   |                   |                   |    |                |    |         |    |          |    |          |       |                   |       |                   |           |     |                   |     |               |     |         |     |              |    |                 |     |                 |     |  |  |
| Error           | Decrement Counter  | Error             | Decrement Counter |    |                |    |         |    |          |    |          |       |                   |       |                   |           |     |                   |     |               |     |         |     |              |    |                 |     |                 |     |  |  |
| CRC Error       | Yes  | Data Buffer Error | Yes               |    |                |    |         |    |          |    |          |       |                   |       |                   |           |     |                   |     |               |     |         |     |              |    |                 |     |                 |     |  |  |
| Timeout Error   | Yes  | Stalled           | No*               |    |                |    |         |    |          |    |          |       |                   |       |                   |           |     |                   |     |               |     |         |     |              |    |                 |     |                 |     |  |  |
| NAK Received    | No   | Bit stuff Error   | Yes               |    |                |    |         |    |          |    |          |       |                   |       |                   |           |     |                   |     |               |     |         |     |              |    |                 |     |                 |     |  |  |
| Babble Detected | No*  |                   |                   |    |                |    |         |    |          |    |          |       |                   |       |                   |           |     |                   |     |               |     |         |     |              |    |                 |     |                 |     |  |  |
| 26              | <p><b>Low Speed Device (LS).</b> This bit indicates that the target device (USB data source or sink) is a low speed device, running at 1.5 Mb/s, instead of at full speed (12 Mb/sec). There are special restrictions on schedule placement for low speed TDs. If an ICH root hub port is connected to a full speed device and this bit is set to a 1 for a low speed transaction, the ICH sends out a low speed preamble on that port before sending the PID. No preamble is sent if a ICH root hub port is connected to a low speed device.</p> <p>1 = Low Speed Device<br/>0 = Full Speed Device</p>  |                   |                   |    |                |    |         |    |          |    |          |       |                   |       |                   |           |     |                   |     |               |     |         |     |              |    |                 |     |                 |     |  |  |
| 25              | <p><b>Isochronous Select (IOS).</b> The field specifies the type of the data structure. If this bit is set to a 1, then the TD is an isochronous transfer. Isochronous TDs are always marked inactive by the hardware after execution, regardless of the results of the transaction.</p> <p>1 = Isochronous Transfer Descriptor<br/>0 = Non-isochronous Transfer Descriptor</p>  |                   |                   |    |                |    |         |    |          |    |          |       |                   |       |                   |           |     |                   |     |               |     |         |     |              |    |                 |     |                 |     |  |  |
| 24              | <p><b>Interrupt on Complete (IOC).</b> This specifies that the ICH should issue an interrupt on completion of the frame in which this Transfer Descriptor is executed. Even if the Active bit in the TD is already cleared when the TD is fetched (no transaction will occur on USB), an IOC interrupt is generated at the end of the frame.</p> <p>1 = Issue IOC</p>  |                   |                   |    |                |    |         |    |          |    |          |       |                   |       |                   |           |     |                   |     |               |     |         |     |              |    |                 |     |                 |     |  |  |
| 23              | <p><b>Active.</b> For ICH schedule execution operations, see the Data Transfers To/From Main Memory section.</p> <p>1 = Set to 1 by software to enable the execution of a message transaction by the ICH.<br/>0 = When the transaction associated with this descriptor is completed, the ICH sets this bit to 0 indicating that the descriptor should not be executed when it is next encountered in the schedule. The Active bit is also set to 0 if a stall handshake is received from the endpoint.</p>   |                   |                   |    |                |    |         |    |          |    |          |       |                   |       |                   |           |     |                   |     |               |     |         |     |              |    |                 |     |                 |     |  |  |

Table 5-56. TD Control and Status (Sheet 2 of 2)

| Bit   | Description   |
|-------|---|
| 22    | <p><b>Stalled.</b></p> <p>1 = Set to a 1 by the ICH during status updates to indicate that a serious error has occurred at the device/endpoint addressed by this TD. This can be caused by babble, the error counter counting down to zero, or reception of the STALL handshake from the device during the transaction. Any time that a transaction results in the Stalled bit being set, the Active bit is also cleared (set to 0). If a STALL handshake is received from a SETUP transaction, a Time Out Error will also be reported.</p>   |
| 21    | <p><b>Data Buffer Error (DBE).</b></p> <p>1 = Set to a 1 by the ICH during status update to indicate that the ICH is unable to keep up with the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (underrun). When this occurs, the actual length and Max Length field of the TD will not match. In the case of an underrun, the ICH will transmit an incorrect CRC (thus invalidating the data at the endpoint) and leave the TD active (unless error count reached zero). If an overrun condition occurs, the ICH will force a timeout condition on the USB, invalidating the transaction at the source.</p>   |
| 20    | <p><b>Babble Detected (BABD).</b></p> <p>1 = Set to a 1 by the ICH during status update when "babble" is detected during the transaction generated by this descriptor. Babble is unexpected bus activity for more than a preset amount of time. In addition to setting this bit, the ICH also sets the "STALLED" bit (bit 22) to a 1. Since "babble" is considered a fatal error for that transfer, setting the "STALLED" bit to a 1 insures that no more transactions occur as a result of this descriptor. Detection of babble causes immediate termination of the current frame. No further TDs in the frame are executed. Execution resumes with the next frame list index.</p>   |
| 19    | <p><b>Negative Acknowledgment (NAK) Received (NAKR).</b></p> <p>1 = Set to a 1 by the ICH during status update when the ICH receives a "NAK" packet during the transaction generated by this descriptor. If a NAK handshake is received from a SETUP transaction, a Time Out Error will also be reported.</p>   |
| 18    | <p><b>CRC/Time Out Error (CRC_TOUT).</b></p> <p>1 = Set to a 1 by the ICH as follows:</p> <ul style="list-style-type: none"> <li>• During a status update in the case that no response is received from the target device/endpoint within the time specified by the protocol chapter of the USB specification.</li> <li>• During a status update when a Cyclical Redundancy Check (CRC) error is detected during the transaction associated with this transfer descriptor.</li> <li>• In the transmit case (OUT or SETUP Command), this is in response to the ICH detecting a timeout from the target device/endpoint.</li> <li>• In the receive case (IN Command), this is in response to the ICH's CRC checker circuitry detecting an error on the data received from the device/endpoint or a NAK or STALL handshake being received in response to a SETUP transaction.</li> </ul> |
| 17    | <p><b>Bit stuff Error (BSE).</b></p> <p>1 = This bit is set to a 1 by the ICH during status update to indicate that the receive data stream contained a sequence of more than 6 ones in a row.</p>  |
| 16    | <p><b>Bus Turn Around Time-out (BTTO).</b></p> <p>1 = This bit is set to a 1 by the ICH during status updates to indicate that a bus time-out condition was detected for this USB transaction. This time-out is specially defined as not detecting an IDLE-to 'K' state Start of Packet (SOP) transition from 16 to 18 bit times after the SE0-to IDE transition of previous End of Packet (EOP).</p>   |
| 15:11 | Reserved  |
| 10:0  | <p><b>Actual Length (ACTLEN).</b> The Actual Length field is written by the ICH at the conclusion of a USB transaction to indicate the actual number of bytes that were transferred. It can be used by the software to maintain data integrity. The value programmed in this register is encoded as n-1 (see Maximum Length field description in the TD Token).</p>   |

Table 5-57. TD Token

| Bit   | Description  |
|-------|--|
| 31:21 | <p><b>Maximum Length (MAXLEN).</b> The Maximum Length field specifies the maximum number of data bytes allowed for the transfer. The Maximum Length value does not include protocol bytes, such as Packet ID (PID) and CRC. The maximum data packet is 1280 bytes. The 1280 packet length is the longest packet theoretically guaranteed to fit into a frame. Actual packet maximum lengths are set by HCD according to the type and speed of the transfer. Note that the maximum length allowed by the USB specification is 1023 bytes. The valid encodings for this field are:</p> <p>0x000 = 1 byte<br/>                     0x001 = 2 bytes<br/>                     ....<br/>                     0x3FE = 1023 bytes<br/>                     0x3FF = 1024 bytes<br/>                     ....<br/>                     0x4FF = 1280 bytes<br/>                     0x7FF = 0 bytes (null data packet)</p> <p>Note that values from 500h to 7FEh are illegal and cause a consistency check failure.</p> <p>In the transmit case, the ICH uses this value as a terminal count for the number of bytes it fetches from host memory. In most cases, this is the number of bytes it will actually transmit. In rare cases, the ICH may be unable to access memory (e.g., due to excessive latency) in time to avoid underrunning the transmitter. In this instance the ICH would transmit fewer bytes than specified in the Maximum Length field.</p> |
| 20    | Reserved.  |
| 19    | <p><b>Data Toggle (D).</b> This bit is used to synchronize data transfers between a USB endpoint and the host. This bit determines which data PID is sent or expected (0=DATA0 and 1=DATA1). The Data Toggle bit provides a 1-bit sequence number to check whether the previous packet completed. This bit must always be 0 for Isochronous TDs.</p>   |
| 18:15 | <p><b>Endpoint (ENDPT).</b> This 4-bit field extends the addressing internal to a particular device by providing 16 endpoints. This permits more flexible addressing of devices in which more than one sub-channel is required.</p>  |
| 14:8  | <p><b>Device Address.</b> This field identifies the specific device serving as the data source or sink.</p>  |
| 7:0   | <p><b>Packet Identification (PID).</b> This field contains the Packet ID to be used for this transaction. Only the IN (69h), OUT (E1h), and SETUP (2Dh) tokens are allowed. Any other value in this field causes a consistency check failure resulting in an immediate halt of the ICH. Bits [3:0] are complements of bits [7:4].</p>  |

Table 5-58. TD Buffer Pointer

| Bit  | Description  |
|------|--|
| 31:0 | <p><b>Buffer Pointer (BUFF_PNT).</b> Bits [31:0] corresponds to memory address [31:0], respectively. It points to the beginning of the buffer that will be used during this transaction. This buffer must be at least as long as the value in the Maximum Length field described in the TD Token. The data buffer may be byte-aligned.</p> |

### 5.15.1.3 Queue Head (QH)

Queue heads are special structures used to support the requirements of Control, Bulk, and Interrupt transfers. Since these TDs are not automatically retired after each use, their maintenance requirements can be reduced by putting them into a queue. Queue Heads must be aligned on a 16-byte boundary, and the elements are shown in Table 5-59.

**Table 5-59. Queue Head Block**

| Bytes | Description                | Attributes |
|-------|----------------------------|------------|
| 00-03 | Queue Head Link Pointer    | RO         |
| 04-07 | Queue Element Link Pointer | R/W        |

**Table 5-60. Queue Head Link Pointer**

| Bit  | Description   |
|------|---|
| 31:4 | <b>Queue Head Link Pointer (QHLP).</b> This field contains the address of the next data object to be processed in the horizontal list and corresponds to memory address signals [31:4], respectively.   |
| 3:2  | Reserved. These bits must be written as 0s.   |
| 1    | <b>QH/TD Select (Q).</b> This bit indicates to the hardware whether the item referenced by the link pointer is another TD or a QH.<br>1=QH<br>0=TD  |
| 0    | <b>Terminate (T).</b> This bit indicates to the ICH that this is the last QH in the schedule. If there are active TDs in this queue, they are the last to be executed in this frame.<br>1 = Last QH (pointer is invalid).<br>0 = Pointer is valid (points to a QH or TD). |

**Table 5-61. Queue Element Link Pointer**

| Bit  | Description  |
|------|--|
| 31:4 | <b>Queue Element Link Pointer (QELP).</b> This field contains the address of the next TD or QH to be processed in this queue and corresponds to memory address signals [31:4], respectively.   |
| 3:2  | Reserved.  |
| 1    | <b>QH/TD Select (Q).</b> This bit indicates to the hardware whether the item referenced by the link pointer is another TD or a QH. For entries in a queue, this bit is typically set to 0.<br>1 = QH<br>0 = TD   |
| 0    | <b>Terminate (T).</b> This bit indicates to the ICH that there are no valid TDs in this queue. When HCD has new queue entries it overwrites this value with a new TD pointer to the queue entry.<br>1 = Terminate (No valid queue entries).<br>0 = Pointer is valid. |

## 5.15.2 Data Transfers To/From Main Memory

The following sections describe the details on how HCD and the ICH communicate via the Schedule data structures. The discussion is organized in a top-down manner, beginning with the basics of walking the Frame List, followed by a description of generic processing steps common to all transfer descriptors, and finally a discussion on Transfer Queuing.

### 5.15.2.1 Executing the Schedule

Software programs the ICH with the starting address of the Frame List and the Frame List index, then causes the ICH to execute the schedule by setting the Run/Stop bit in the Control register to Run. The ICH processes the schedule one entry at a time: the next element in the frame list is not fetched until the current element in the frame list is retired.

Schedule execution proceeds in the following fashion:

- The ICH first fetches an entry from the Frame List. This entry has three fields. Bit 0 indicates whether the address pointer field is valid. Bit 1 indicates whether the address points to a Transfer Descriptor or to a queue head. The third field is the pointer itself.
- If isochronous traffic is to be moved in a given frame, the Frame List entry points to a Transfer Descriptor. If no isochronous data is to be moved in that frame, the entry points to a queue head or the entry is marked invalid and no transfers are initiated in that frame.
- If the Frame List entry indicates that it points to a Transfer Descriptor, the ICH fetches the entry and begins the operations necessary to initiate a transaction on USB. Each TD contains a link field that points to the next entry, as well as indicating whether it is a TD or a QH.
- If the Frame List entry contains a pointer to a QH, the ICH processes the information from the QH to determine the address of the next data object that it should process.
- The TD/QH process continues until the millisecond allotted to the current frame expires. At this point, the ICH fetches the next entry from the Frame List. If the ICH is not able to process all of the transfer descriptors during a given frame, those descriptors are retired by software without having been executed.

### 5.15.2.2 Processing Transfer Descriptors

The ICH executes a TD using the following generalized algorithm. These basic steps are common across all modes of TDs. Subsequent sections present processing steps unique to each TD mode.

1. ICH Fetches TD or QH from the current Link Pointer.
2. If a QH, go to 1 to fetch from the Queue Element Link Pointer. If inactive, go to 12
3. Build Token, actual bits are in TD Token.
4. If (Host-to-Function) then
  - [*PCI Access*] issue request for data, (referenced through TD.BufferPointer)
  - wait for first chunk data arrival
  - end if
5. [*Begin USB Transaction*] Issue Token (from token built in 2, above) and begin data transfer.
  - if (Host-to-Function) then Go to 6
  - else Go to 7
  - end if
6. Fetch data from memory (via TD BufferPointer) and transfer over USB until TD Max-Length bytes have been read and transferred. [*Concurrent system memory and USB Accesses*].
  - Go to 8.

7. Wait for data to arrive (from USB). Write incoming bytes into memory beginning at TD BufferPointer. Internal HC buffer should signal end of data packet. Number of bytes received must be (TD Max-Length; The length of the memory area referenced by TD BufferPointer must be (TD Max-Length. [*Concurrent system memory and USB Accesses*]).
8. Issue handshake based on status of data received (Ack or Time-out). Go to 10.
9. Wait for handshake, if required [*End of USB Transaction*].
10. Update Status [*PCI Access*] (TD.Status and TD.ActualLength).  
If the TD was an isochronous TD, mark the TD inactive. Go to 12.  
If not an isochronous TD, and the TD completed successfully, mark the TD inactive. Go to 11.  
If not successful, and the error count has not been reached, leave the TD active. If the error count has been reached, mark the TD inactive. Go to 12.
11. Write the link pointer from the current TD into the element pointer field of the QH structure. If the Vf bit is set in the TD link pointer, go to 2.
12. Proceed to next entry.

### 5.15.2.3 Command Register, Status Register, and TD Status Bit Interaction

**Table 5-62. Command Register, Status Register, and TD Status Bit Interaction**

| Condition                                    | ICH USB Status Register Actions  | TD Status Register Actions   |
|--|--|--|
| CRC/Time Out Error                           | <ul style="list-style-type: none"> <li>• Set USB Error Int bit<sup>1</sup>, Clear HC Halted bit</li> </ul>   | <ul style="list-style-type: none"> <li>• Clear Active bit<sup>1</sup> and set Stall bit<sup>1</sup></li> </ul> |
| Illegal PID, PID Error, Max Length (illegal) | <ul style="list-style-type: none"> <li>• Clear Run/Stop bit in command register</li> <li>• Set HC Process Error and HC Halted bits</li> </ul>  |  |
| PCI Master/Target Abort                      | <ul style="list-style-type: none"> <li>• Clear Run/Stop bit in command register</li> <li>• Set Host System Error and HC Halted bits</li> </ul>   |  |
| Suspend Mode                                 | <ul style="list-style-type: none"> <li>• Clear Run/Stop bit in command register<sup>2</sup></li> <li>• Set HC Halted bit</li> </ul>  |  |
| Resume Received and Suspend Mode = 1         | <ul style="list-style-type: none"> <li>• Set Resume received bit</li> </ul>  |  |
| Run/Stop = 0                                 | <ul style="list-style-type: none"> <li>• Clear Run/Stop bit in command register</li> <li>• Set HC Halted bit</li> </ul>  |  |
| Config Flag Set                              | <ul style="list-style-type: none"> <li>• Set Config Flag in command register</li> </ul>  |  |
| HC Reset/Global Reset                        | <ul style="list-style-type: none"> <li>• Clear Run/Stop and Config Flag in command register</li> <li>• Clear USB Int, USB Error Int, Resume received, Host System Error, HC Process Error, and HC Halted bits</li> </ul> |  |
| IOC = 1 in TD Status                         | <ul style="list-style-type: none"> <li>• Set USB Int bit</li> </ul>  |  |
| Stall  | <ul style="list-style-type: none"> <li>• Set USB Error Int bit</li> </ul>  | <ul style="list-style-type: none"> <li>• Clear Active bit<sup>1</sup> and set Stall bit</li> </ul>             |
| Bit Stuff/Data Buffer Error                  | <ul style="list-style-type: none"> <li>• Set USB Error Int bit<sup>1</sup></li> </ul>  | <ul style="list-style-type: none"> <li>• Clear Active bit<sup>1</sup> and set Stall bit<sup>1</sup></li> </ul> |
| Short Packet Detect                          | <ul style="list-style-type: none"> <li>• Set USB Int bit</li> </ul>  | <ul style="list-style-type: none"> <li>• Clear Active bit</li> </ul>   |

**NOTES:**

1. Only If error counter counted down from 1 to 0
2. Suspend mode can be entered only when Run/Stop bit is 0



**Note:** If a NAK or STALL response is received from a SETUP transaction, a Time Out Error will be reported. This will cause the Error counter to decrement and the CRC/Time-out Error status bit to be set within the TD Control and Status DWORD during write back. If the Error counter changes from 1 to 0, the Active bit will be reset to 0 and Stalled bit to 1 as normal.

### 5.15.2.4 Transfer Queuing

Transfer Queues are used to implement a guaranteed data delivery stream to a USB Endpoint. Transfer Queues are composed of two parts: a Queue Header (QH) and a linked list. The linked list of TDs and QHs has an indeterminate length (0 to n).

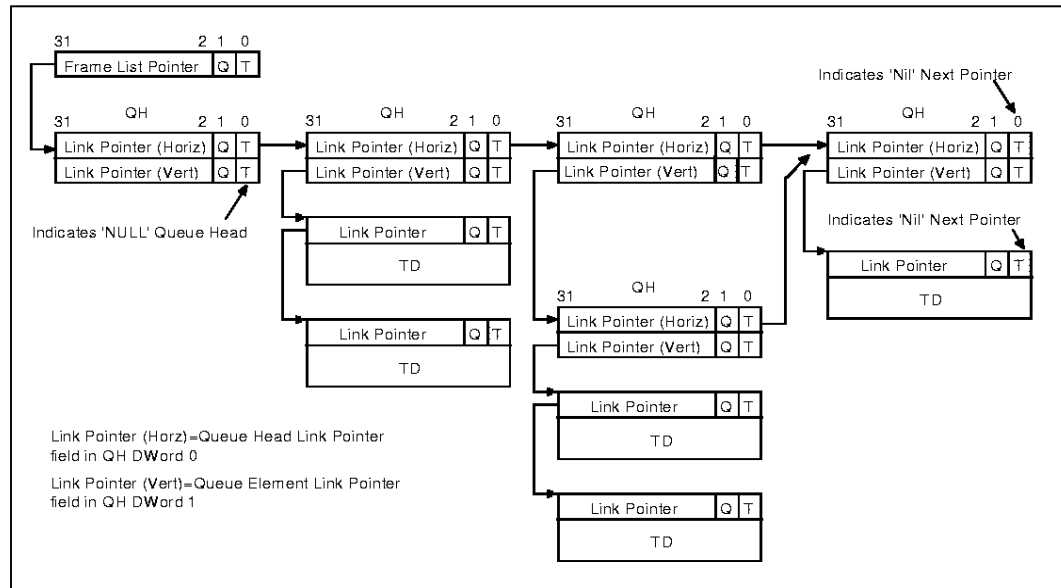
The QH contains two link pointers and is organized as two contiguous dWords. The first dWord is a horizontal pointer (Queue Head Link Pointer), used to link a single transfer queue with either another transfer queue, or a TD (target data structure depends on Q bit). If the T bit is set, this QH represents the last data structure in the current Frame. The T bit informs the ICH that no further processing is required until the beginning of the next frame. The second dWord is a vertical pointer (Queue Element Link Pointer) to the first data structure (TD or QH) being managed by this QH. If the T bit is set, the queue is empty. This pointer may reference a TD or another QH.

Figure 5-14 illustrates four example queue conditions. The first QH (on far left) is an example of an “empty” queue; the termination bit (T Bit), in the vertical link pointer field, is set to 1. The horizontal link pointer references another QH. The next queue is the expected typical configuration. The horizontal link pointer references another QH, and the vertical link pointer references a valid TD.

Typically, the vertical pointer in a QH points to a TD. However, as shown in Figure 5-14 (third example from left side of figure) the vertical pointer could point to another QH. When this occurs, a new Q Context is entered and the Q Context just exited is NULL (ICH will not update the vertical pointer field).

The far right QH is an example of a frame ‘termination’ node. Since its horizontal link pointer has its termination bit set, the ICH assumes there is no more work to complete for the current Frame.

Figure 5-14. Example Queue Conditions



Transfer Queues are based on the following characteristics:

- A QH’s vertical link pointer (Queue Element Link Pointer) references the ‘Top’ queue member. A QH’s horizontal link pointer (Queue Head Link Pointer) references the “next” work element in the Frame.
- Each queue member’s link pointer references the next element within the queue.

In the simplest model, the ICH follows vertical link point to a queue element, then executes the element. If the completion status of the TD satisfies the advance criteria as shown in Table 5-63, the ICH advances the queue by writing the just-executed TD’s link pointer back into the QH’s Queue Element link pointer. The next time the queue head is traversed, the next queue element will be the Top element.

The traversal has two options: Breadth first, or Depth first. A flag bit in each TD (Vf - Vertical Traversal Flag) controls whether traversal is Breadth or Depth first. The default mode of traversal is Breadth-First. For Breadth-First, the ICH only executes the top element from each queue. The execution path is shown below:

1. QH (Queue Element Link Pointer)
2. TD
3. Write-Back to QH (Queue Element Link Pointer)
4. QH (Queue Head Link pointer).

Breadth-First is also performed for every transaction execution that fails the advance criteria. This means that if a queued TD fails, the queue does not advance, and the ICH traverses the QH’s Queue Head Link Pointer.

In a Depth-first traversal, the top queue element must complete successfully to satisfy the *advance criteria* for the queue. If the ICH is currently processing a queue, and the advance criteria are met, and the Vf bit is set, the ICH follows the TD’s link pointer to the next schedule work item.

Note that regardless of traversal model, when the advance criteria are met, the successful TD’s link pointer is written back to the QH’s Queue Element link pointer.

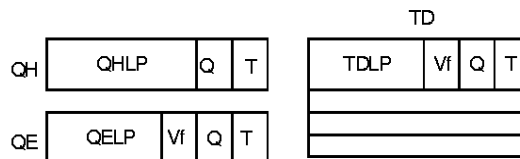
When the ICH encounters a QH, it caches the QH internally, and sets internal state to indicate it is in a Q-context. It needs this state to update the correct QH (for auto advancement) and also to make the correct decisions on how to traverse the Frame List.

Restricting the advancement of queues to advancement criteria implements a guaranteed data delivery stream. A queue is **never** advanced on an error completion status (even in the event the error count was exhausted). Table 5-63 lists the general queue advance criteria, which are based on the execution status of the TD at the ‘Top’ of a currently ‘active’ queue.

**Table 5-63. Queue Advance Criteria**

| Function-to-Host (IN) |           |                 | Host-to-Function (OUT) |           |                 |
|-----------------------|-----------|-----------------|------------------------|-----------|-----------------|
| Non-NULL              | NULL      | Error/NAK       | Non-NULL               | NULL      | Error/NAK       |
| Advance Q             | Advance Q | Retry Q Element | Advance Q              | Advance Q | Retry Q Element |

Table 5-63 is a decision table illustrating the valid combinations of link pointer bits and the valid actions taken when advancement criteria for a queued transfer descriptor are met. The column headings for the link pointer fields are encoded, based on the following list:



Legends:

QH.LP = Queue Head Link Pointer (or Horizontal Link Pointer)  
 QE.LP = Queue Element Link Pointer (or Vertical Link Pointer)  
 TD.LP = TD Link Pointer  
 QH.Q = Q bit in QH  
 QH.T = T bit in QH  
 QE.Q = Q bit in QE  
 QE.T = T bit in QE  
 TD.Vf = Vf bit in TD  
 TD.Q = Q bit in TD  
 TD.T = T bit in TD

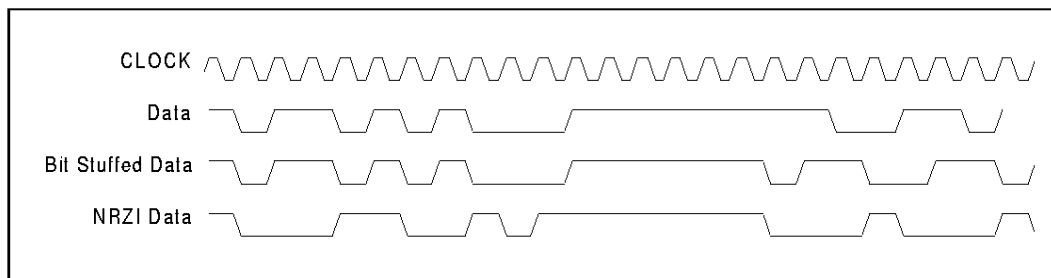
**Table 5-64. USB Schedule List Traversal Decision Table**

| Q Context | QH.Q | QH.T | QE.Q | QE.T | TD.Vf | TD.Q | TD.T | Description  |
|-----------|------|------|------|------|-------|------|------|--|
| 0         | -    | -    | -    | -    | x     | 0    | 0    | Not in Queue - execute TD. Use TD.LP to get next TD.   |
| 0         | -    | -    | -    | -    | x     | x    | 1    | Not in Queue - execute TD. End of Frame  |
| 0         | -    | -    | -    | -    | x     | 1    | 0    | Not in Queue - execute TD. Use TD.LP to get next (QH+QE). Set Q Context to 1.                      |
| 1         | 0    | 0    | 0    | 0    | 0     | x    | x    | In Queue. Use QE.LP to get TD. execute TD. Update QE.LP with TD.LP. Use QH.LP to get next TD.      |
| 1         | x    | x    | 0    | 0    | 1     | 0    | 0    | In Queue. Use QE.LP to get TD. execute TD. Update QE.LP with TD.LP. Use TD.LP to get next TD.      |
| 1         | x    | x    | 0    | 0    | 1     | 1    | 0    | In Queue. Use QE.LP to get TD. execute TD. Update QE.LP with TD.LP. Use TD.LP to get next (QH+QE). |
| 1         | 0    | 0    | x    | 1    | x     | x    | x    | In Queue. Empty queue. Use QH.LP to get next TD  |
| 1         | x    | x    | 1    | 0    | -     | -    | -    | In Queue. Use QE.LP to get (QH+QE)   |
| 1         | x    | 1    | 0    | 0    | 0     | x    | x    | In Queue. Use QE.LP to get TD. execute TD. Update QE.LP with TD.LP. End of Frame                   |
| 1         | x    | 1    | x    | 1    | x     | x    | x    | In Queue. Empty queue. End of Frame  |
| 1         | 1    | 0    | 0    | 0    | 0     | x    | x    | In Queue. Use QE.LP to get TD. execute TD. Update QE.LP with TD.LP. Use QH.LP to get next (QH+QE). |
| 1         | 1    | 0    | x    | 1    | x     | x    | x    | In Queue. Empty queue. Use QH.LP to get next (QH+QE)   |

### 5.15.3 Data Encoding and Bit Stuffing

The USB employs NRZI data encoding (Non-Return to Zero Inverted) when transmitting packets. In NRZI encoding, a 1 is represented by no change in level and a 0 is represented by a change in level. A string of zeros causes the NRZI data to toggle each bit time. A string of ones causes long periods with no transitions in the data. To ensure adequate signal transitions, bit stuffing is employed by the transmitting device when sending a packet on the USB. A 0 is inserted after every six consecutive 1's in the data stream before the data is NRZI encoded to force a transition in the NRZI data stream. This gives the receiver logic a data transition at least once every seven bit times to guarantee the data and clock lock. A waveform of the data encoding is shown in Figure 5-15.

Figure 5-15. USB Data Encoding



Bit stuffing is enabled beginning with the Sync Pattern and throughout the entire transmission. The data “one” that ends the Sync Pattern is counted as the first one in a sequence. Bit stuffing is always enforced, without exception. If required by the bit stuffing rules, a zero bit will be inserted even if it is the last bit before the end-of-packet (EOP) signal.

### 5.15.4 Bus Protocol

#### 5.15.4.1 Bit Ordering

Bits are sent out onto the bus least significant bit (LSb) first, followed by next LSb, through to the most significant bit (MSb) last.

#### 5.15.4.2 SYNC Field

All packets begin with a synchronization (SYNC) field, which is a coded sequence that generates a maximum edge transition density. The SYNC field appears on the bus as IDLE followed by the binary string “KJKJKJKK,” in its NRZI encoding. It is used by the input circuitry to align incoming data with the local clock and is defined to be eight bits in length. SYNC serves only as a synchronization mechanism and is not shown in the following packet diagrams. The last two bits in the SYNC field are a marker that is used to identify the first bit of the PID. All subsequent bits in the packet must be indexed from this point.

### 5.15.4.3 Packet Field Formats

Field formats for the token, data, and handshake packets are described in the following section. The effects of NRZI coding and bit stuffing have been removed for the sake of clarity. All packets have distinct start and end of packet delimiters.

**Table 5-65. PID Format**

| Bit | Data Sent | Bit | Data Sent  |
|-----|-----------|-----|------------|
| 0   | PID 0     | 4   | NOT(PID 0) |
| 1   | PID 1     | 5   | NOT(PID 1) |
| 2   | PID 2     | 6   | NOT(PID 2) |
| 3   | PID 3     | 7   | NOT(PID 3) |

#### Packet Identifier Field

A packet identifier (PID) immediately follows the SYNC field of every USB packet. A PID consists of a four bit packet type field followed by a four-bit check field as shown in Table 5-65. The PID indicates the type of packet and, by inference, the format of the packet and the type of error detection applied to the packet. The four-bit check field of the PID insures reliable decoding of the PID so that the remainder of the packet is interpreted correctly. The PID check field is generated by performing a ones complement of the packet type field.

Any PID received with a failed check field or which decodes to a non-defined value is assumed to be corrupted and the remainder of the packet is assumed to be corrupted and is ignored by the receiver. PID types, codes, and descriptions are listed in Table 5-66.

**Table 5-66. PID Types**

| PID Type  | PID Name | PID[3:0] | Description   |
|-----------|----------|----------|---|
| Token     | OUT      | b0001    | Address + endpoint number in host -> function transaction                                 |
|           | IN       | b1001    | Address + endpoint number in function -> host transaction                                 |
|           | SOF      | b0101    | Start of frame marker and frame number  |
| Data      | SETUP    | b1101    | Address + endpoint number in host -> function transaction for setup to a control endpoint |
|           | DATA0    | b0011    | Data packet PID even  |
|           | DATA1    | b1011    | Data packet PID odd   |
| Handshake | ACK      | b0010    | Receiver accepts error free data packet   |
|           | NAK      | b1010    | Rx device cannot accept data or Tx device cannot send data                                |
|           | STALL    | b1110    | Endpoint is stalled   |
| Special   | PRE      | b1100    | Host-issued preamble. Enables downstream bus traffic to low speed devices.                |

PIDs are divided into four coding groups: token, data, handshake, and special, with the first two transmitted PID bits (PID[1:0]) indicating which group. This accounts for the distribution of PID codes.

#### 5.15.4.4 Address Fields

Function endpoints are addressed using two fields: the function address field and the endpoint field.

**Table 5-67. Address Field**

| Bit | Data Sent | Bit | Data Sent |
|-----|-----------|-----|-----------|
| 0   | ADDR 0    | 4   | ADDR 4    |
| 1   | ADDR 1    | 5   | ADDR 5    |
| 2   | ADDR 2    | 6   | ADDR 6    |
| 3   | ADDR 3    |     |           |

##### Address Field

The function address (ADDR) field specifies the function, via its address, that is either the source or destination of a data packet, depending on the value of the token PID. As shown in Table 5-67, a total of 128 addresses are specified as ADDR[6:0]. The ADDR field is specified for IN, SETUP, and OUT tokens.

##### Endpoint Field

An additional four-bit endpoint (ENDP) field, shown in Table 5-68, permits more flexible addressing of functions in which more than one sub-channel is required. Endpoint numbers are function specific. The endpoint field is defined for IN, SETUP, and OUT token PIDs only.

**Table 5-68. Endpoint Field**

| Bit | Data Sent |
|-----|-----------|
| 0   | ENDP 0    |
| 1   | ENDP 1    |
| 2   | ENDP 2    |
| 3   | ENDP 3    |

#### 5.15.4.5 Frame Number Field

The frame number field is an 11-bit field that is incremented by the host on a per frame basis. The frame number field rolls over upon reaching its maximum value of x7FF, and is sent only for SOF tokens at the start of each frame.

#### 5.15.4.6 Data Field

The data field may range from 0 to 1023 bytes and must be an integral numbers of bytes. Data bits within each byte are shifted out LSB first.

#### 5.15.4.7 Cyclic Redundancy Check (CRC)

CRC is used to protect the all non-PID fields in token and data packets. In this context, these fields are considered to be protected fields. The PID is not included in the CRC check of a packet containing CRC. All CRCs are generated over their respective fields in the transmitter before bit stuffing is performed. Similarly, CRCs are decoded in the receiver after stuffed bits have been

removed. Token and data packet CRCs provide 100% coverage for all single and double bit errors. A failed CRC is considered to indicate that one or more of the protected fields is corrupted and causes the receiver to ignore those fields, and, in most cases, the entire packet.

## 5.15.5 Packet Formats

### 5.15.5.1 Token Packets

Table 5-69 shows the field formats for a token packet. A token consists of a PID, specifying either IN, OUT, or SETUP packet type, and ADDR and ENDP fields. For OUT and SETUP transactions, the address and endpoint fields uniquely identify the endpoint that will receive the subsequent data packet. For IN transactions, these fields uniquely identify which endpoint should transmit a data packet. Only the ICH can issue token packets. IN PIDs define a data transaction from a function to the ICH. OUT and SETUP PIDs define data transactions from the ICH to a function.

Token packets have a five-bit CRC which covers the address and endpoint fields as shown above. The CRC does not cover the PID, which has its own check field. Token and SOF packets are delimited by an EOP after three bytes of packet field data. If a packet decodes as an otherwise valid token or SOF but does not terminate with an EOP after three bytes, it must be considered invalid and ignored by the receiver.

**Table 5-69. Token Format**

| Packet | Width  |
|--------|--------|
| PID    | 8 bits |
| ADDR   | 7 bits |
| ENDP   | 4 bits |
| CRC5   | 5 bits |

### 5.15.5.2 Start of Frame Packets

Table 5-70 shows a start of frame (SOF) packet. SOF packets are issued by the host at a nominal rate of once every 1.00 ms 0.05. SOF packets consist of a PID indicating packet type followed by an 11-bit frame number field.

The SOF token comprises the token-only transaction that distributes a start of frame marker and accompanying frame number at precisely timed intervals corresponding to the start of each frame. All full speed functions, including hubs, must receive and decode the SOF packet. The SOF token does not cause any receiving function to generate a return packet; therefore, SOF delivery to any given function cannot be guaranteed. The SOF packet delivers two pieces of timing information. A function is informed that a start of frame has occurred when it detects the SOF PID. Frame timing sensitive functions, which do not need to keep track of frame number, need only decode the SOF PID; they can ignore the frame number and its CRC. If a function needs to track frame number, it must comprehend both the PID and the time stamp.

**Table 5-70. SOF Packet**

| Packet       | Width   |
|--------------|---------|
| PID          | 8 bits  |
| Frame Number | 11 bits |
| CRC5         | 5 bits  |

### 5.15.5.3 Data Packets

A data packet consists of a PID, a data field, and a CRC as shown in Table 5-71. There are two types of data packets, identified by differing PIDs: DATA0 and DATA1. Two data packet PIDs are defined to support data toggle synchronization.

Data must always be sent in integral numbers of bytes. The data CRC is computed over only the data field in the packet and does not include the PID, which has its own check field.

Table 5-71. Data Packet Format

| Packet | Width        |
|--------|--------------|
| PID    | 8 bits       |
| DATA   | 0-1023 bytes |
| CRC16  | 16 bits      |

### 5.15.5.4 Handshake Packets

Handshake packets consist of only a PID. Handshake packets are used to report the status of a data transaction and can return values indicating successful reception of data, flow control, and stall conditions. Only transaction types that support flow control can return handshakes. Handshakes are always returned in the handshake phase of a transaction and may be returned, instead of data, in the data phase. Handshake packets are delimited by an EOP after one byte of packet field. If a packet is decoded as an otherwise valid handshake but does not terminate with an EOP after one byte, it must be considered invalid and ignored by the receiver.

There are three types of handshake packets:

- **ACK** indicates that the data packet was received without bit stuff or CRC errors over the data field and that the data PID was received correctly. An ACK handshake is applicable only in transactions in which data has been transmitted and where a handshake is expected. ACK can be returned by the host for IN transactions and by a function for OUT transactions.
- **NAK** indicates that a function was unable to accept data from the host (OUT) or that a function has no data to transmit to the host (IN). NAK can only be returned by functions in the data phase of IN transactions or the handshake phase of OUT transactions. The host can never issue a NAK. NAK is used for flow control purposes to indicate that a function is temporarily unable to transmit or receive data, but will eventually be able to do so without need of host intervention. NAK is also used by interrupt endpoints to indicate that no interrupt is pending.
- **STALL** is returned by a function in response to an IN token or after the data phase of an OUT. STALL indicates that a function is unable to transmit or receive data, and that the condition requires host intervention to remove the stall. Once a function's endpoint is stalled, the function must continue returning STALL until the condition causing the stall has been cleared through host intervention. The host is not permitted to return a STALL under any condition.

### 5.15.5.5 Handshake Responses

#### IN Transaction

A function may respond to an IN transaction with a STALL or NAK. If the token received was corrupted, the function will issue no response. If the function can transmit data, it will issue the data packet. The ICH, as the USB host, can return only one type of handshake on an IN transaction, an ACK. If it receives a corrupted data, or cannot accept data due to a condition such as an internal buffer overrun, it discards the data and issues no response.



### OUT Transaction

A function may respond to an OUT transaction with a STALL, ACK, or NAK. If the transaction contained corrupted data, it will issue no response.

### SETUP Transaction

Setup defines a special type of host to function data transaction which permits the host to initialize an endpoint's synchronization bits to those of the host. Upon receiving a Setup transaction, a function must accept the data. Setup transactions cannot be STALLED or NAKed and the receiving function must accept the Setup transfer's data. If a non-control endpoint receives a SETUP PID, it must ignore the transaction and return no response.

## 5.15.6 USB Interrupts

There are two general groups of USB interrupt sources, those resulting from execution of transactions in the schedule, and those resulting from an ICH operation error. All transaction-based sources can be masked by software through the ICH's Interrupt Enable register. Additionally, individual transfer descriptors can be marked to generate an interrupt on completion.

When the ICH drives an interrupt for USB, it drives the PIRQD# pin active for interrupts occurring due to ports 0 and 1 until all sources of the interrupt are cleared.

### 5.15.6.1 Transaction Based Interrupts

These interrupts are not signaled until after the status for the last complete transaction in the frame has been written back to host memory. This guarantees that software can safely process through (Frame List Current Index -1) when it is servicing an interrupt.

#### CRC Error / Time-out

A CRC/Time-out error occurs when a packet transmitted from the ICH to a USB device or a packet transmitted from a USB device to the ICH generates a CRC error. The ICH is informed of this event by a time-out from the USB device or by the ICH's CRC checker generating an error on reception of the packet. Additionally, a USB bus time-out occurs when USB devices do not respond to a transaction phase within 19 bit times of an EOP. Either of these conditions will cause the C\_ERR field of the TD to decrement.

When the C\_ERR field decrements to zero, the following occurs:

- The Active bit in the TD is cleared
- The Stalled bit in the TD is set
- The CRC/Time-out bit in the TD is set.
- At the end of the frame, the USB Error Interrupt bit is set in the HC status register.

If the CRC/Time out interrupt is enabled in the Interrupt Enable register, a hardware interrupt will be signaled to the system.

### **Interrupt on Completion**

Transfer Descriptors contain a bit that can be set to cause an interrupt on their completion. The completion of the transaction associated with that block causes the USB Interrupt bit in the HC Status Register to be set at the end of the frame in which the transfer completed. When a TD is encountered with the IOC bit set to 1, the IOC bit in the HC Status register is set to 1 at the end of the frame if the active bit in the TD is set to 0 (even if it was set to zero when initially read).

If the IOC Enable bit of Interrupt Enable register (bit 2 of I/O offset 04h) is set, a hardware interrupt is signaled to the system. The USB Interrupt bit in the HC status register is set either when the TD completes successfully or because of errors. If the completion is because of errors, the USB Error bit in the HC status register is also set.

### **Short Packet Detect**

A transfer set is a collection of data which requires more than 1 USB transaction to completely move the data across the USB. An example might be a large print file which requires numerous TDs in multiple frames to completely transfer the data. Reception of a data packet that is less than the endpoint's Max Packet size during Control, Bulk or Interrupt transfers signals the completion of the transfer set, even if there are active TDs remaining for this transfer set. Setting the SPD bit in a TD indicates to the HC to set the USB Interrupt bit in the HC status register at the end of the frame in which this event occurs. This feature streamlines the processing of input on these transfer types. If the Short Packet Interrupt Enable bit in the Interrupt Enable register is set, a hardware interrupt is signaled to the system at the end of the frame where the event occurred.

### **Serial Bus Babble**

When a device transmits on the USB for a time greater than its assigned Max Length, it is said to be babbling. Since isochrony can be destroyed by a babbling device, this error results in the Active bit in the TD being cleared to 0 and the Stalled and Babble bits being set to one. The C\_ERR field is not decremented for a babble. The USB Error Interrupt bit in the HC Status register is set to 1 at the end of the frame. A hardware interrupt is signaled to the system.

If an EOF babble was caused by the ICH (due to incorrect schedule for instance), the ICH will force a bit stuff error followed by an EOP and the start of the next frame.

### **Stalled**

This event indicates that a device/endpoint returned a STALL handshake during a transaction or that the transaction ended in an error condition. The TDs Stalled bit is set and the Active bit is cleared. Reception of a STALL does not decrement the error counter. A hardware interrupt is signaled to the system.

### **Data Buffer Error**

This event indicates that an overrun of incoming data or a under-run of outgoing data has occurred for this transaction. This would generally be caused by the ICH not being able to access required data buffers in memory within necessary latency requirements. Either of these conditions will cause the C\_ERR field of the TD to be decremented.

When C\_ERR decrements to zero, the Active bit in the TD is cleared, the Stalled bit is set, the USB Error Interrupt bit in the HC Status register is set to 1 at the end of the frame and a hardware interrupt is signaled to the system.

### Bit Stuff Error

A bit stuff error results from the detection of a sequence of more than 6 ones in a row within the incoming data stream. This will cause the C\_ERR field of the TD to be decremented. When the C\_ERR field decrements to zero, the Active bit in the TD is cleared to 0, the Stalled bit is set to one, the USB Error Interrupt bit in the HC Status register is set to 1 at the end of the frame and a hardware interrupt is signaled to the system.

## 5.15.6.2 Non-Transaction Based Interrupts

If an ICH process error or system error occur, the ICH halts and immediately issues a hardware interrupt to the system.

### Resume Received

This event indicates that the ICH received a RESUME signal from a device on the USB bus during a global suspend. If this interrupt is enabled in the Interrupt Enable register, a hardware interrupt will be signaled to the system allowing the USB to be brought out of the suspend state and returned to normal operation.

### ICH Process Error

The HC monitors certain critical fields during operation to ensure that it does not process corrupted data structures. These include checking for a valid PID and verifying that the MaxLength field is less than 1280. If it detects a condition that would indicate that it is processing corrupted data structures, it immediately halts processing, sets the HC Process Error bit in the HC Status register and signals a hardware interrupt to the system.

This interrupt cannot be disabled through the Interrupt Enable register.

### Host System Error

The ICH sets this bit to 1 when a PCI Parity error, PCI Master Abort, or PCI Target Abort occur. When this error occurs, the ICH clears the Run/Stop bit in the Command register to prevent further execution of the scheduled TDs. This interrupt cannot be disabled through the Interrupt Enable register.

## 5.15.7 USB Power Management

The Host Controller can be put into a suspended state and its power can be removed. This requires that certain bits of information are retained in the resume power plane of the ICH so that a device on a port may wake the system. Such a device may be a fax-modem, which will wake up the machine to receive a fax or take a voice message. The following bits in I/O space are to be maintained when the ICH enters a low power state:

**Table 5-72. Bits to be maintained in low power states**

| Register                | Offset    | Bit | Description                      |
|-------------------------|-----------|-----|----------------------------------|
| Command                 | 00h       | 3   | Enter Global Suspend Mode (EGSM) |
| Status                  | 02h       | 2   | Resume Detect                    |
| Port Status and Control | 10h & 12h | 2   | Port Enabled/Disabled            |
|                         |           | 6   | Resume Detect                    |
|                         |           | 8   | Low Speed Device Attached        |
|                         |           | 12  | Suspend                          |

When the ICH detects a resume event on any of its ports, it will set the USB\_STS bit in ACPI space. If USB is enabled as a wake/break event, the system will wake up and an SCI will be generated.

## 5.15.8 USB Legacy Keyboard Operation

When a USB keyboard is plugged into the system, and a standard keyboard is not, the system may not boot, and DOS legacy software will not run, because the keyboard will not be identified. The ICH implements a series of trapping operations which will snoop accesses that go to the keyboard controller, and put the expected data from the USB keyboard into the keyboard controller.

**Note:** The scheme described below assumes that the keyboard controller (8042 or equivalent) is on the LPC bus.

This legacy operation is performed through SMM space.

Figure 5-16 shows the Enable and Status path. The latched SMI source (60R, 60W, 64R, 64W) is available in the Status Register. Because the enable is after the latch, it is possible to check for other events that didn't necessarily cause an SMI. It is the software's responsibility to logically AND the value with the appropriate enable bits.

Note also that the SMI is generated before the PCI cycle completes (e.g., before TRDY# goes active) to ensure that the processor does not complete the cycle before the SMI is observed.

The logic will also need to block the accesses to the 8042. If there is an external 8042, then this is simply accomplished by not activating the 8042 CS. This is simply done by logically ANDing the 4 enables (60R, 60W, 64R, 64W) with the 4 types of accesses to determine if 8042CS should go active. An additional term is required for the "Pass-through" case. The state table for the diagram is shown in Table 5-73.

Figure 5-16. USB Legacy Keyboard Flow Diagram

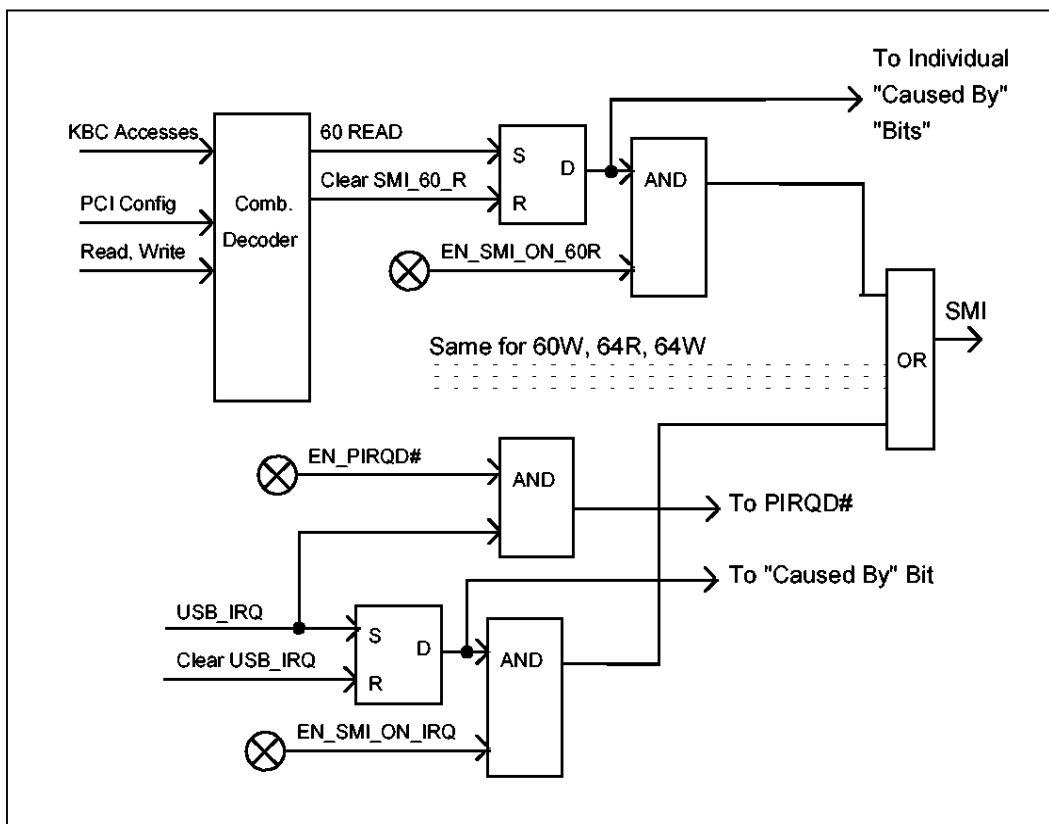


Table 5-73. USB Legacy Keyboard State Transitions

| Current State | Action      | Data Value | Next State  | Comment  |
|---------------|-------------|------------|-------------|--|
| IDLE          | 64h / Write | D1h        | GateState 1 | Standard D1 command. Cycle passed through to 8042. SMI# doesn't go active. PSTATE goes to 1.   |
| IDLE          | 64h / Write | Not D1h    | IDLE        | Bit 3 in Config Register determines if cycle passed through to 8042 and if SMI# generated.   |
| IDLE          | 64h / Read  | N/A        | IDLE        | Bit 2 in Config Register determines if cycle passed through to 8042 and if SMI# generated.   |
| IDLE          | 60h / Write | Don't Care | IDLE        | Bit 1 in Config Register determines if cycle passed through to 8042 and if SMI# generated.   |
| IDLE          | 60h / Read  | N/A        | IDLE        | Bit 0 in Config Register determines if cycle passed through to 8042 and if SMI# generated.   |
| GateState 1   | 60h / Write | XXh        | GateState 2 | Cycle passed through to 8042, even if trap enabled in Bit 1 in Config Register. No SMI# generated. PSTATE remains 1. If data value is not DFh or DDh then the 8042 may chose to ignore it.                   |
| GateState 1   | 64h / Write | D1h        | GateState 1 | Cycle passed through to 8042, even if trap enabled via Bit 3 in Config Register. No SMI# generated. PSTATE remains 1. Stay in GateState1 because this is part of the double-trigger sequence.                |
| GateState 1   | 64h / Write | Not D1h    | IDLE        | Bit 3 in Config space determines if cycle passed through to 8042 and if SMI# generated. PSTATE goes to 0. If Bit 7 in Config Register is set, then SMI# should be generated.                                 |
| GateState 1   | 60h / Read  | N/A        | IDLE        | This is an invalid sequence. Bit 0 in Config Register determines if cycle passed through to 8042 and if SMI# generated. PSTATE goes to 0. If Bit 7 in Config Register is set, then SMI# should be generated. |
| GateState 1   | 64h / Read  | N/A        | GateState 1 | Just stay in same state. Generate an SMI# if enabled in Bit 2 of Config Register. PSTATE remains 1.  |
| GateState 2   | 64 / Write  | FFh        | IDLE        | Standard end of sequence. Cycle passed through to 8042. PSTATE goes to 0. Bit 7 in Config Space determines if SMI# should be generated.  |
| GateState 2   | 64h / Write | Not FFh    | IDLE        | Improper end of sequence. Bit 3 in Config Register determines if cycle passed through to 8042 and if SMI# generated. PSTATE goes to 0. If Bit 7 in Config Register is set, then SMI# should be generated.    |
| GateState 2   | 64h / Read  | N/A        | GateState 2 | Just stay in same state. Generate an SMI# if enabled in Bit 2 of Config Register. PSTATE remains 1.  |
| GateState 2   | 60h / Write | XXh        | IDLE        | Improper end of sequence. Bit 1 in Config Register determines if cycle passed through to 8042 and if SMI# generated. PSTATE goes to 0. If Bit 7 in Config Register is set, then SMI# should be generated.    |
| GateState 2   | 60h / Read  | N/A        | IDLE        | Improper end of sequence. Bit 0 in Config Register determines if cycle passed through to 8042 and if SMI# generated. PSTATE goes to 0. If Bit 7 in Config Register is set, then SMI# should be generated.    |

## 5.16 SMBus Controller Functional Description (D31:F3)

The ICH provides an SMBus host controller including an SMBus Master interface. The host controller provides a mechanism for the processor to initiate communications with SMBus peripherals (slaves). The ICH is also capable of operating in a mode in which it can communicate with I<sup>2</sup>C compatible devices.

The ICH SMBus logic exists in Device 31:Function 3 configuration space, and consists of a transmit data path, and host controller. The transmit data path provides the data flow logic needed to implement the seven different SMBus command protocols and is controlled by the host controller. The ICH SMBus controller logic is clocked by RTC clock.

The programming model of the host controller is combined into two portions: a PCI configuration portion, and a system I/O mapped portion. All static configuration, such as the I/O base address, is done via the PCI configuration space. Real-time programming of the Host interface is done in system I/O space.

### 5.16.1 Host Controller

The SMBus Host Controller is used to send commands to other SMBus slave devices. Software sets up the host controller with an address, command, and, for writes, data, and then tells the controller to start. When the controller has finished transmitting data on writes, or receiving data on reads, it will generate an SMI# or interrupt, if enabled.

The host controller supports 7 command protocols of the SMBus interface (see System Management Bus Specs, Rev 1.0): Quick Command, Send Byte, Receive Byte, Write Byte/Word, Read Byte/Word, Process Call, and Block Read/Write.

The SMBus Host Controller requires that the various data and command fields be setup for the type of command to be sent. When software sets the START bit, the SMBus Host Controller will perform the requested transaction, and interrupt the processor (or generate an SMI#) when the transaction is completed. Once a START command has been issued, the values of the “active registers” (Host Control, Host Command, Transmit Slave Address, Data 0, Data 1) should not be changed or read until the interrupt status bit (INTR) has been set (indicating the completion of the command). Any register values needed for computation purposes should be saved prior to issuing of a new command, as the SMBus Host Controller will update all registers while completing the new command.

#### 5.16.1.1 Command Protocols

In all of the following commands, the Host Status Register (offset 00h) is used to determine the progress of the command. While the command is in operation, the HOST\_BUSY bit is set. If the command completes successfully, and the INTREN bit is set in the Host Control Register, the INTR bit will be set in the Host Status Register. If the device does not respond with an acknowledge, and the transaction times out, the DEV\_ERR bit is set. If software sets the KILL bit in the Host Control Register while the command is running, the transaction will stop and the FAILED bit will be set.

**Quick Command**

When programmed for a Quick Command, the Transmit Slave Address Register is sent. The format of the protocol is shown in Table 5-74.

**Table 5-74. Quick Protocol**

| Bit | Description            |
|-----|------------------------|
| 1   | Start Condition        |
| 2-8 | Slave Address - 7 bits |
| 9   | Read / Write Direction |
| 10  | Acknowledge from slave |
| 11  | Stop                   |

**Send Byte / Receive Byte**

For the Send Byte command, the Transmit Slave Address and Device Command Registers are sent. For the Receive Byte command, the Transmit Slave Address Register is sent. The data received is stored in the DATA0 register.

The Receive Byte is similar to a Send Byte, the only difference is the direction of data transfer. The format of the protocol is shown in Table 5-75.

**Table 5-75. Send / Receive Byte Protocol**

| Send Byte Protocol |                        | Receive Byte Protocol |                        |
|--------------------|------------------------|-----------------------|------------------------|
| Bit                | Description            | Bit                   | Description            |
| 1                  | Start                  | 1                     | Start                  |
| 2-8                | Slave Address - 7 bits | 2-8                   | Slave Address - 7 bits |
| 9                  | Write                  | 9                     | Read                   |
| 10                 | Acknowledge from slave | 10                    | Acknowledge from slave |
| 11-18              | Command code - 8 bits  | 11-18                 | Data byte from slave   |
| 19                 | Acknowledge from slave | 19                    | NOT Acknowledge        |
| 20                 | Stop                   | 20                    | Stop                   |



### Write Byte/Word

The first byte of a Write Byte/Word access is the command code. The next 1 or 2 bytes are the data to be written. When programmed for a write byte/word command, the Transmit Slave Address, Device Command, and Data0 Registers are sent. In addition, the Data1 Register is sent on a write word command. The format of the protocol is shown in Table 5-76.

**Table 5-76. Write Byte/Word Protocol**

| Write Byte Protocol |                        | Write Word Protocol |                         |
|---------------------|------------------------|---------------------|-------------------------|
| Bit                 | Description            | Bit                 | Description             |
| 1                   | Start                  | 1                   | Start                   |
| 2-8                 | Slave Address - 7 bits | 2-8                 | Slave Address - 7 bits  |
| 9                   | Write                  | 9                   | Write                   |
| 10                  | Acknowledge from slave | 10                  | Acknowledge from slave  |
| 11-18               | Command code - 8 bits  | 11-18               | Command code - 8 bits   |
| 19                  | Acknowledge from slave | 19                  | Acknowledge from slave  |
| 20-27               | Data Byte - 8 bits     | 20-27               | Data Byte Low - 8 bits  |
| 28                  | Acknowledge from Slave | 28                  | Acknowledge from Slave  |
| 29                  | Stop                   | 29-36               | Data Byte High - 8 bits |
|                     |                        | 37                  | Acknowledge from slave  |
|                     |                        | 38                  | Stop                    |

### Read Byte/Word

Reading data is slightly more complicated than writing data. First the ICH must write a command to the slave device. Then it must follow that command with a repeated start condition to denote a read from that device's address. The slave then returns 1 or 2 bytes of data.

When programmed for the read byte/word command, the Transmit Slave Address and Device Command Registers are sent. Data is received into the DATA0 on the read byte, and the DATA0 and DATA1 registers on the read word. The format of the protocol is shown in Table 5-77.

**Table 5-77. Read Byte/Word Protocol**

| Read Byte Protocol |                          | Read Word Protocol |                                    |
|--------------------|--------------------------|--------------------|------------------------------------|
| Bit                | Description              | Bit                | Description                        |
| 1                  | Start                    | 1                  | Start                              |
| 2-8                | Slave Address - 7 bits   | 2-8                | Slave Address - 7 bits             |
| 9                  | Write                    | 9                  | Write                              |
| 10                 | Acknowledge from slave   | 10                 | Acknowledge from slave             |
| 11-18              | Command code - 8 bits    | 11-18              | Command code - 8 bits              |
| 19                 | Acknowledge from slave   | 19                 | Acknowledge from slave             |
| 20                 | Repeated Start           | 20                 | Repeated Start                     |
| 21-27              | Slave Address - 7 bits   | 21-27              | Slave Address - 7 bits             |
| 28                 | Read                     | 28                 | Read                               |
| 29                 | Acknowledge from slave   | 29                 | Acknowledge from slave             |
| 30-37              | Data from slave - 8 bits | 30-37              | Data Byte Low from slave - 8 bits  |
| 38                 | NOT acknowledge          | 38                 | Acknowledge                        |
| 39                 | Stop                     | 39-46              | Data Byte High from slave - 8 bits |
|                    |                          | 47                 | NOT acknowledge                    |
|                    |                          | 48                 | Stop                               |

### Process Call

The process call is so named because a command sends data and waits for the slave to return a value dependent on that data. The protocol is simply a Write Word followed by a Read Word, but without a second command or stop condition.

When programmed for the Process Call command, the ICH transmits the Transmit Slave Address, Host Command, DATA0 and DATA1 registers. Data received from the device is stored in the DATA0 and DATA1 registers. The format of the protocol is shown in Table 5-78.

**Note:** For process call command, the value written into bit 0 of the Transmit Slave Address Register (SMB I/O register, offset 04h) needs to be 0.

**Table 5-78. Process Call Protocol**

| Bit   | Description                        |
|-------|------------------------------------|
| 1     | Start                              |
| 2-8   | Slave Address - 7 bits             |
| 9     | Write                              |
| 10    | Acknowledge from Slave             |
| 11-18 | Command code - 8 bits              |
| 19    | Acknowledge from slave             |
| 20-27 | Data byte Low - 8 bits             |
| 28    | Acknowledge from slave             |
| 29-36 | Data Byte High - 8 bits            |
| 37    | Acknowledge from slave             |
| 38    | Repeated Start                     |
| 39-45 | Slave Address - 7 bits             |
| 46    | Read                               |
| 47    | Acknowledge from slave             |
| 48-55 | Data Byte Low from slave - 8 bits  |
| 56    | Acknowledge                        |
| 57-64 | Data Byte High from slave - 8 bits |
| 65    | NOT acknowledge                    |
| 66    | Stop                               |

### Block Read/Write

The Block Write begins with a slave address and a write condition. After the command code, the ICH issues a byte count which describes how many more bytes will follow in the message. If a slave had 20 bytes to send, the first byte would be the number 20 (14h), followed by the 20 bytes of data. The byte count may not be 0.

Instead of a 32-byte buffer for Block Read/Write command, the ICH implements the Block Data Byte register (D31:F3, I/O offset 07h) for Block Read/Write command.

When programmed for a block write command, the Transmit Slave Address, Host Command, and Data0 (count) registers are sent. Data is then sent from the Block Data Byte register. After the byte has been sent, the ICH will set the BYTE\_DONE\_STS bit in the Host Status register. If there are more bytes to send, the software will write the next byte to the Block Data Byte register and will also clear the BYTE\_DONE\_STS bit. The ICH will then send the next byte.

On block read commands, after the byte count is stored in the DATA 0 register, the first data byte goes in the Block Data Byte register; the ICH will then set the BYTE\_DONE\_STS bit and generate an SMI# or interrupt. The SMI# or interrupt handler will read the byte and then clear the BYTE\_DONE\_STS bit to allow the next byte to be read into the Block Data Byte register. Note that after receiving data byte N-1 of the block, the software needs to set the LAST\_BYTE bit in the Host Control Register; this allows the ICH to send a NOT ACK (instead of an ACK) after receiving the last data byte (byte N) of the block.

After each byte of a block message, the ICH sets the BYTE\_DONE\_STS bit and generates an interrupt or SMI#. Software clears the BYTE\_DONE\_STS bit before the next transfer occurs. When the interrupt handler clears the BYTE\_DONE\_STS bit after the last byte has been transferred, the ICH sets the INTR bit and generates another interrupt to signal the end of the block transfer. Thus, for a block message of n bytes, the ICH will generate n+1 interrupts. The interrupt handler needs to be implemented to handle all of these interrupts.

The format of the Block Read/Write protocol is shown in Table 5-79.

**Note:** For Block Write, if the I<sup>2</sup>C\_EN bit is set, the format of the command changes slightly. The ICH will still send the number of bytes indicated in the DATA0 register. However, it will not send the contents of the Data 0 register as part of the message.

Table 5-79. Block Read/Write Protocol

| Block Write Protocol |  | Block Read Protocol |                                   |
|----------------------|--|---------------------|-----------------------------------|
| Bit                  | Description  | Bit                 | Description                       |
| 1                    | Start  | 1                   | Start                             |
| 2-8                  | Slave Address - 7 bits   | 2-8                 | Slave Address - 7 bits            |
| 9                    | Write  | 9                   | Write                             |
| 10                   | Acknowledge from slave   | 10                  | Acknowledge from slave            |
| 11-18                | Command code - 8 bits  | 11-18               | Command code - 8 bits             |
| 19                   | Acknowledge from slave   | 19                  | Acknowledge from slave            |
| 20-27                | Byte Count - 8 bits<br>(Skip this step if I <sup>2</sup> C_En bit set) | 20                  | Repeated Start                    |
| 28                   | Acknowledge from Slave<br>(Skip this step if I2C_EN bit set)           | 21-27               | Slave Address - 7 bits            |
| 29-36                | Data Byte 1 - 8 bits   | 28                  | Read                              |
| 37                   | Acknowledge from Slave   | 29                  | Acknowledge from slave            |
| 38-45                | Data Byte 2 - 8 bits   | 30-37               | Byte Count from slave - 8 bits    |
| 46                   | Acknowledge from slave   | 38                  | Acknowledge                       |
| ...                  | Data Bytes / Slave<br>Acknowledges...                                  | 39-46               | Data Byte 1 from slave - 8 bits   |
| ...                  | Data Byte N - 8 bits   | 47                  | Acknowledge                       |
| ...                  | Acknowledge from Slave   | 48-55               | Data Byte 2 from slave - 8 bits   |
| ...                  | Stop   | 56                  | Acknowledge                       |
|                      |  | ...                 | Data Bytes from slave/Acknowledge |
|                      |  | ...                 | Data Byte N from slave - 8 bits   |
|                      |  | ...                 | NOT Acknowledge                   |
|                      |  | ...                 | Stop                              |

### I<sup>2</sup>C Read

This command allows the ICH to perform block reads to certain I<sup>2</sup>C devices, such as serial E<sup>2</sup>PROMs. The SMBus Block Read sends both the 7-bit address, as well as the Command field. This command field could be used as the extended 10-bit address for accessing I<sup>2</sup>C devices that use 10-bit addressing.

However, this doesn't allow access to devices using the I<sup>2</sup>C "Combined Format" that has data bytes after the address. Typically, these data bytes correspond to an offset (address) within the serial memory chips.

**Note:** This new command is supported independent of the setting of the I2C\_EN bit.

For I<sup>2</sup>C Read command, the value written into bit 0 of the Transmit Slave Address Register (SMB I/O register, offset 04h) needs to be 0.

The format that is used for the new command is shown in Table 5-80:

Table 5-80. I<sup>2</sup>C Block Read

| Bit   | Description                         |
|-------|-------------------------------------|
| 1     | Start                               |
| 2-8   | Slave Address - 7 bits              |
| 9     | Write                               |
| 10    | Acknowledge from slave              |
| 11-18 | Command code - 8 bits               |
| 19    | Acknowledge from slave              |
| 20-27 | Send DATA0 register                 |
| 28    | Acknowledge from slave              |
| 29-36 | Send DATA1 register                 |
| 37    | Acknowledge from slave              |
| 38    | Repeated start                      |
| 39-45 | Slave Address - 7 bits              |
| 46    | Read                                |
| 47    | Acknowledge from slave              |
| 48-55 | Data byte from slave                |
| 56    | Acknowledge                         |
| 57-64 | Data byte 2 from slave - 8 bits     |
| 65    | Acknowledge                         |
| -     | Data bytes from slave / Acknowledge |
| -     | Data byte N from slave - 8 bits     |
| -     | NOT Acknowledge                     |
| -     | Stop                                |

The ICH will continue reading data from the peripheral until the NAK is received.

### 5.16.1.2 I<sup>2</sup>C Behavior

When the I<sup>2</sup>C\_EN bit is set, the ICH SMBus logic will instead be set to communicate with I<sup>2</sup>C devices. This forces the following changes:

1. The Process Call command will skip the Command code (and its associated acknowledge)
2. The Block Write command will skip sending the Byte Count (DATA0)

In addition, the ICH will support the new I<sup>2</sup>C Read command. This is independent of the I<sup>2</sup>C\_EN bit.

## 5.16.2 Bus Arbitration

Several masters may attempt to get on the bus at the same time by driving the SMBDATA line low to signal a start condition. The ICH must continuously monitor the SMBDATA line. When the ICH is attempting to drive the bus to a '1' by letting go of the SMBDATA line, and it samples SMBDATA low, then some other master is driving the bus and the ICH must stop transferring data.

If the ICH sees that it has lost arbitration, the condition is called a collision. The ICH will set the BUS\_ERR bit in the Host Status Register, and if enabled, generate an interrupt or SMI#. The processor is responsible for restarting the transaction.

When the ICH is a SMBus master, it will drive the clock. When the ICH is sending address or command as an SMBus master, or data bytes as a master on writes, it will drive data relative to the clock it is also driving. It will not start toggling the clock until the start or stop condition meets proper setup and hold time. The ICH will also guarantee minimum time between SMBus transactions as a master.

### Clock Stretching

Some devices may not be able to handle their clock toggling at the rate that the ICH as an SMBus master would like. They have the capability of stretching the low time of the clock. When the ICH attempts to release the clock (allowing the clock to go high), the clock will remain low for an extended period of time.

The ICH must monitor the SMBus clock line after it releases the bus to determine whether to enable the counter for the high time of the clock. While the bus is still low, the high time counter must not be enabled. Similarly, the low period of the clock can be stretched by an SMBus master if it is not ready to send or receive data.

The ICH never stretches the low period of the clock. It will always have the data to transfer on writes, and it should always have a spot for the data on reads.

### Bus Time Out (ICH as SMBus Master)

If there is an error in the transaction, such that an SMBus device does not signal an acknowledge, or holds the clock lower than the allowed time-out time, the transaction will time out. The ICH will discard the cycle, and set the DEV\_ERR bit. The time out minimum is 25 ms. The time-out counter inside the ICH will start after the first bit of data is transferred by the ICH and it is waiting for a response. The 25 ms will be a count of 800 RTC clocks. The time-out counter will not count when the BYTE\_DONE\_STS bit (SMBus I/O offset 00h, bit 7) is set and the SECOND\_TO\_STS bit (TCO I/O offset 06h, bit 1) is not set.

## 5.16.3 Interrupts / SMI#

The ICH SMBus controller uses PIRQB# as its interrupt pin. However, the system can alternatively be set up to generate SMI# instead of an interrupt, by setting the SMBUS\_SMI\_EN bit.

## 5.16.4 SMBALERT#

SMBALERT# is multiplexed with GPIO[11]. When enabled and the signal is asserted, the ICH can generate an interrupt, an SMI# or a wake event from S1-S4.

## 5.17 ICH AC'97 2.1 Functional Description

This section describes the AC-link interface. A functional description of the ICH operation is discussed in this section. This includes an overview of the AC-link implementation in the ICH, as well as a detailed feature list of what is supported. Also included in this section is a recommended feature list for AC'97 codecs. These are features that Intel recommends for a codec designed for the ICH AC'97 controller.

For further information on the operation of the AC-link protocol, refer to the AC'97 specification. References to AC'97 in this document refer to the AC'97 2.1 specification.

The ICH supports the following features:

- Independent PCI functions for audio and modem
- Independent channels for PCM in and PCM out, microphone in
- Left and right audio channels
- Single modem line
- 16 bit sample resolution
- Multiple sample rates
- 16 GPIOs
- Dual codec configuration with two SDIN pins

Table 5-81 shows a detailed list of features supported by the ICH AC'97 digital controller.

**Table 5-81. Features Supported by ICH**

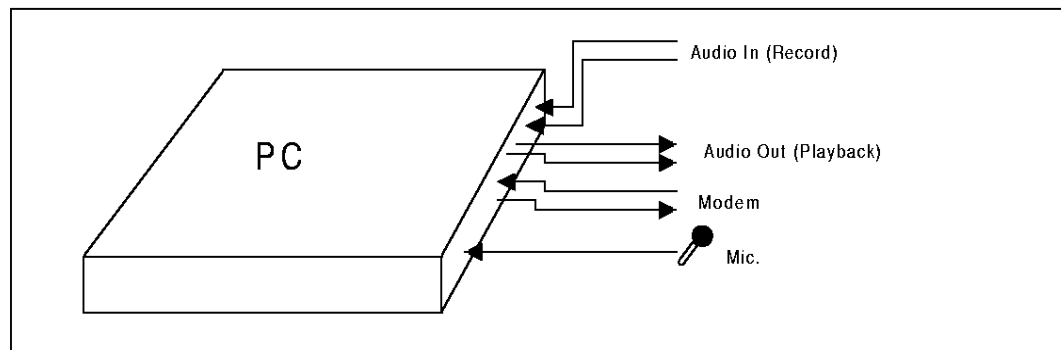
| Feature            | Description   |
|--------------------|---|
| System Interface   | <ul style="list-style-type: none"> <li>• Isochronous low latency bus master memory interface</li> <li>• Scatter/gather support for word-aligned buffers in memory (all mono or stereo 16-bit data types are supported, no 8-bit data types are supported)</li> <li>• Data buffer size in system memory from 3 to 65535 samples per input</li> <li>• Data buffer size in system memory from 0 to 65535 samples per output</li> <li>• Independent PCI audio and modem functions with configuration and IO spaces</li> <li>• AC'97 codec registers are shadowed in system memory via driver (not PCI IO space)</li> <li>• AC'97 codec register accesses are serialized via semaphore bit in PCI IO space (new accesses are not allowed while a prior access is still in progress)</li> </ul> |
| Power Management   | <ul style="list-style-type: none"> <li>• Power management via ACPI control methods</li> <li>Support for audio states: <i>D0</i>, <i>D2</i>, <i>D3hot</i>, <i>D3cold</i></li> <li>Support for modem states: <i>D0</i>, <i>D3hot</i>, <i>D3cold</i></li> <li>• SCI event generation for PCI modem function with wake-up from <i>D3cold</i></li> <li>• Independent codec D3 w/ Link down event, synchronized via two bit semaphore (in PCI IO Space)</li> </ul>  |
| PCI Audio Function | <ul style="list-style-type: none"> <li>• Read/write access to audio codec registers 00h-3Ah and vendor registers 5Ah-7Eh</li> <li>• 16-bit stereo PCM output, up to 48 kHz (L,R channels on slots 3,4)</li> <li>• 16-bit stereo PCM input, up to 48 kHz (L,R channels on slots 3,4)</li> <li>• 16-bit mono mic in w/ or w/o mono mix, up to 48 kHz (L,R channel, slots 3,4) (mono mix supports mono hardware AEC reference for speakerphone)</li> <li>• 16-bit mono PCM input, up to 48 kHz from dedicated mic ADC (slot 6) (supports speech recognition or stereo hardware AEC ref for speakerphone)</li> <li>• During cold reset AC_RST# is held low until after POST and software deassertion of AC_RST# (supports passive PC_BEEP to speaker connection during POST)</li> </ul>       |



**Table 5-81. Featured Supported by ICH**

| Feature            | Description   |
|--------------------|---|
| PCI Modem function | <ul style="list-style-type: none"> <li>• Read/write access to modem codec registers 3Ch-58h and vendor registers 5Ah-7Eh</li> <li>• 16-bit mono modem line1 output and input, up to 48 kHz (slot 5)</li> <li>• Low latency GPIO[15:0] via hardwired update between slot 12 and PCI IO register</li> <li>• Programmable PCI interrupt on modem GPIO input changes via slot 12 GPIO_INT</li> <li>• SCI event generation on primary or secondary SDIN wake-up signal</li> </ul>  |
| AC-link            | <ul style="list-style-type: none"> <li>• AC'97 2.1 compliant AC-link interface</li> <li>• Variable sample rate output support via AC'97 SLOTREQ protocol (slots 3,4,5)</li> <li>• Variable sample rate input support via monitoring of slot valid tag bits (slots 3,4,5,6)</li> <li>• 3.3 V digital operation meets AC'97 2.1 DC switching levels</li> <li>• AC-Link IO driver capability meets AC'97 2.1 dual codec specifications</li> <li>• Codec register status reads must be returned with data in the next AC-link frame, per AC'97 2.1 spec.</li> </ul> |
| Multiple Codec     | <ul style="list-style-type: none"> <li>• Dual codec addressing: All AC'97 codec register accesses are addressable to codec ID 00 (primary) or codec ID 01 (secondary)</li> <li>• Dual codec receive capability via primary and secondary SDIN pins (primary, secondary SDIN frames are internally validated, synch'd, and OR'd)</li> </ul>  |

Note that throughout this document references to D31:F5 indicate that the audio function exists in PCI device 31, function 5. References to D31:F6 indicate that the modem function exists in PCI device 31, function 5.

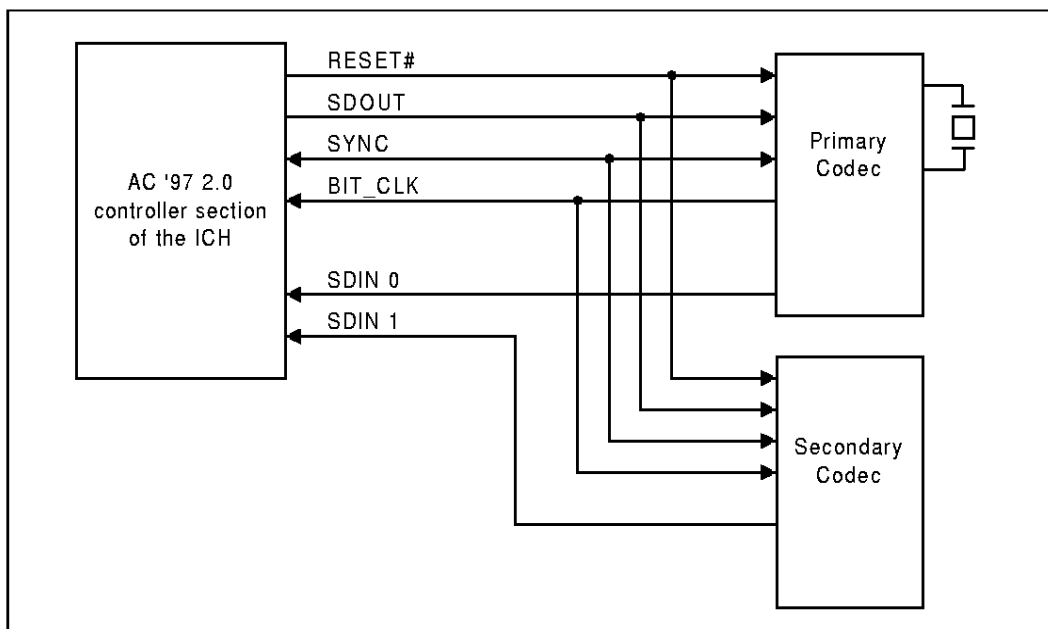
**Figure 5-17. ICH Based AC'97 2.1**


### 5.17.1 AC-link Overview

The ICH is an AC'97 2.1 compliant controller that communicates with companion codecs via a digital serial link called the AC-link. All digital audio/modem streams and command/status information is communicated over the AC-link.

The AC-link is a bi-directional, serial PCM digital stream. It handles multiple input and output data streams, as well as control register accesses, employing a time division multiplexed (TDM) scheme. The AC-link architecture provides for data transfer through individual frames transmitted in a serial fashion. Each frame is divided into 12 outgoing and 12 incoming data streams, or slots. The architecture of the ICH AC-link allows a maximum of two codecs to be connected. Figure 5-18 shows a two codec topology of the AC-link for the ICH, in a serial fashion.

Figure 5-18. AC'97 2.1 Controller-Codec Connection



The AC-link consists of a five signal interface between the controller and codec. Table 5-82 indicates the AC-link signal pins on the ICH and their associated power wells.

Table 5-82. AC'97 Signals

| Signal Name | Type   | Power Well* | Description                  |
|-------------|--------|-------------|------------------------------|
| AC_RESET#   | Output | Resume      | Master hardware reset        |
| AC_SYNC     | Output | Core        | 48kHz fixed rate sample sync |
| AC_BIT_CLK  | Input  | Core        | 12.288MHz Serial data clock  |
| AC_SDOUT    | Output | Core        | Serial output data           |
| AC_SDIN 0   | Input  | Resume      | Serial input data            |
| AC_SDIN 1   | Input  | Resume      | Serial input data            |

**NOTE:** Power well voltage levels are 3.3V

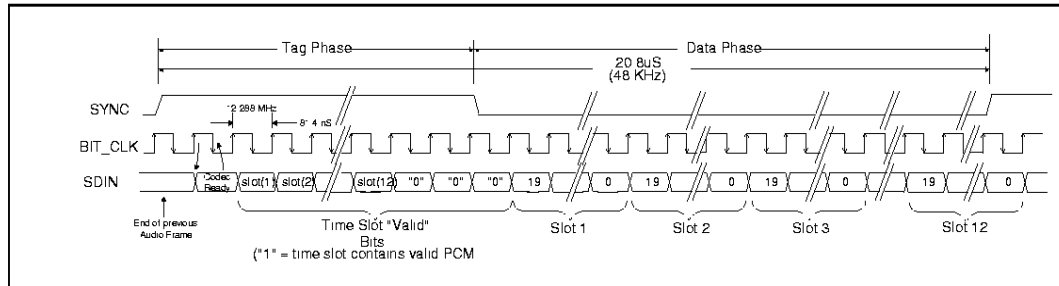
ICH core well outputs may be used as strapping options for the ICH, sampled during system reset. These signals may have weak pullups/pulldowns on them, however this will not interfere with link operation. ICH inputs integrate weak pulldowns to prevent floating traces when a secondary codec is not attached. When the Shut Off bit in the control register is set, all buffers will be turned off and the pins will be held in a steady state, based on these pullups/pulldowns.

BIT\_CLK is fixed at 12.288MHz and is sourced by the primary codec. It provides the necessary clocking to support the twelve 20 bit time slots. AC-link serial data is transitioned on each rising edge of BIT\_CLK. The receiver of AC-link data samples each serial bit on the falling edge of BIT\_CLK.

Synchronization of all AC-link data transactions is signaled by the AC'97 controller via the AC\_SYNC signal, as shown in Figure 5-19. The primary codec drives the serial bit clock onto the AC-link, which the AC'97 controller then qualifies with the AC\_SYNC signal to construct data frames. AC\_SYNC, fixed at 48 KHz, is derived by dividing down BIT\_CLK. AC\_SYNC remains

high for a total duration of 16 BIT\_CLKs at the beginning of each frame. The portion of the frame where AC\_SYNC is high is defined as the tag phase. The remainder of the frame where AC\_SYNC is low is defined as the data phase. Each data bit is sampled on the falling edge of BIT\_CLK.

**Figure 5-19. AC-link Protocol**



The ICH has two SDIN pins allowing a single or dual codec configuration. When two codecs are connected, the primary and secondary codecs can be connected to either SDIN line, however it is recommended that the primary codec be attached to SDIN [0]. The ICH does not distinguish between primary and secondary codecs on its SDIN[1:0] pins, however the registers do distinguish between SDIN[0] and SDIN[1] for wake events, etc. The primary codec can be an AC (audio codec), MC (modem codec), or AMC (audio/modem codec) device. The secondary codec can only be an MC device.

Valid codec configurations include the following:

- AC (Primary)
- MC (Primary)
- AC (Primary) + MC (Secondary)
- AMC (Primary)

The ICH does not support optional test modes as outlined in the AC'97 specification.

## 5.17.2 AC-link Output Frame (SDOUT)

A new audio output frame begins with a low to high transition of AC\_SYNC. AC\_SYNC is synchronous to the rising edge of BIT\_CLK. On the immediately following falling edge of BIT\_CLK, the codec samples the assertion of AC\_SYNC. This falling edge marks the time when both sides of AC-link are aware of the start of a new frame. On the next rising edge of BIT\_CLK, the ICH transitions SDOUT into the first bit position of slot 0, or the valid frame bit. Each new bit position is presented to the AC-link on a rising edge of BIT\_CLK, and subsequently sampled by the codec on the following falling edge of BIT\_CLK. This sequence ensures that data transitions and subsequent sample points for both incoming and outgoing data streams are time aligned.

The output frame data phase corresponds to the multiplexed bundles of all digital output data targeting codec DAC inputs and control registers. Each output frame supports up to twelve outgoing data time slots. The ICH generates 16 bit samples and, in compliance with the AC'97 specification, pads the 4 least significant bits of valid slots with zeros.

The output data stream is sent with the most significant bit first, and all invalid slots are stuffed with zeros. When mono audio sample streams are output from the ICH, software must ensure both left and right sample stream time slots are filled with the same data.

### Output Slot 0: Tag Phase

Slot 0 is considered the tag phase. The tag phase is a special 16 bit time slot where each bit conveys a valid tag for its corresponding time slot within the current frame. A one in a given bit position of slot 0 indicates that the corresponding time slot within the current frame has been assigned to a data stream and contains valid data. If a slot is tagged invalid with a zero in the corresponding bit position of slot 0, the ICH stuffs the corresponding slot with zeros during that slot's active time.

Within slot 0, the first bit is a valid frame bit (slot 0, bit 15) which flags the validity of the entire frame. If the valid frame bit is set to one, this indicates that the current frame contains at least one slot with valid data. When there is no transaction in progress, the ICH will deassert the frame valid bit. Note that after a write to slot 12, that slot will always stay valid, and therefore the frame valid bit will remain set.

The next 12 bit positions of slot 0 (bits [14:3]) indicate which of the corresponding twelve time slots contain valid data. Bits [1:0] of slot 0 are used as codec ID bits to distinguish between separate codecs on the link.

Using the valid bits in the tag phase allows data streams of differing sample rates to be transmitted across the link at its fixed 48 kHz frame rate. The codec can control the output sample rate of the ICH using the SLOTREQ bits as described in the AC'97 specification.

### Output Slot 1: Command Address Port

The command port is used to control features and monitor status of AC'97 functions including, but not limited to, mixer settings and power management. The control interface architecture supports up to 64 16-bit read/write registers, addressable on even byte boundaries. Only the even registers (00h, 02h, etc.) are valid. Output frame slot 1 communicates control register address, and write/read command information.

In the case of the split codec implementation, accesses to the codecs are differentiated by the driver using address offsets 00h - 7Fh for the primary codec and address offsets 80h-FEh for the secondary codec. The differentiation on the link, however, is done via the codec ID bits. See Section 5.1.7.4 for further details.

### Output Slot 2: Command Data Port

The command data port is used to deliver 16-bit control register write data in the event that the current command port operation is a write cycle as indicated in slot 1, bit 19. If the current command port operation is a read, then the entire slot time stuffed with 0's by the ICH. Bits [19:4] contain the write data. Bits [3:0] are reserved and are stuffed with zeros.

### Output Slot 3: PCM Playback Left Channel

Output frame slot 3 is the composite digital audio left playback stream. Typically, this slot is composed of standard PCM (.wav) output samples digitally mixed by the host processor. The ICH transmits sample streams of 16 bits and stuffs the remaining bits with zeros.

Data in output slots 3 and 4 from the ICH should be duplicated by software if there is only a single channel out.

### Output Slot 4: PCM Playback Right Channel

Output frame slot 4 is the composite digital audio right playback stream. Typically, this slot is composed of standard PCM (.wav) output samples digitally mixed by the host processor. The ICH transmits sample streams of 16 bits and stuffs the remaining bits with zeros.

Data in output slots 3 and 4 from the ICH should be duplicated by software if there is only a single channel out.

#### Output Slot 5: Modem Codec

Output frame slot 5 contains modem DAC data. The modem DAC output supports 16 bit resolution. At boot time, if the modem codec is supported, the AC'97 controller driver determines the DAC resolution. During normal runtime operation the ICH stuffs trailing bit positions within this time slot with zeros.

#### Output Slots 6-11: Reserved

Output frame slots 6-11 are reserved and are always stuffed with 0's by the ICH AC'97 controller.

#### Output Slot 12: I/O Control

16 bits of DAA and GPIO control (output) and status (input) have been directly assigned to bits on slot 12 in order to minimize latency of access to changing conditions.

The value of the bits in this slot are the values written to the GPIO control register at offset 54h and D4h (in the case of a secondary codec) in the modem codec I/O space. The following rules govern the usage of slot 12.

1. Slot 12 is marked invalid by default on coming out of AC-link reset, and will remain invalid until a register write to 54h/D4h.
2. A write to offset 54h/D4h in codec I/O space will cause the write data to be transmitted on slot 12 in the next frame, with slot 12 marked valid, and the address/data information to also be transmitted on slots 1 and 2.
3. After the first write to offset 54h/D4h, slot 12 remains valid for all following frames. The data transmitted on slot 12 is the data last written to offset 54h/D4h. Any subsequent write to the register will cause the new data to be sent out on the next frame.
4. Slot 12 will get invalidated after the following events: PCI reset, AC'97 cold reset, warm reset, and hence a wake from S3, S4, or S5. Slot 12 will remain invalid until the next write to offset 54h/D4h.

### 5.17.3 AC-link Input Frame (SDIN)

There are two SDIN lines on the ICH for use with a primary and secondary codec. Each SDIN pin can have a codec attached. Depending upon which codec (AC, MC, or AMC) is attached, various slots will be valid or invalid. The data slots on the two inputs must be completely orthogonal (except for the tag slot 0); that is, no two data slots at the same location will be valid on both lines. This precludes the use of two similar codecs, such as two ACs or MCs, which use the same time slots.

The input frame data streams correspond to the multiplexed bundles of all digital input data targeting the AC'97 controller. As in the case for the output frame, each AC-link input frame consists of twelve time slots.

A new audio input frame begins with a low to high transition of AC\_SYNC. AC\_SYNC is synchronous to the rising edge of BIT\_CLK. On the immediately following falling edge of BIT\_CLK, the receiver samples the assertion of AC\_SYNC. This falling edge marks the time when both sides of AC-link are aware of the start of a new audio frame. On the next rising edge of BIT\_CLK, the codec transitions SDIN into the first bit position of slot 0 (codec ready bit). Each

new bit position is presented to AC-link on a rising edge of BIT\_CLK, and subsequently sampled by the ICH on the following falling edge of BIT\_CLK. This sequence ensures that data transitions and subsequent sample points for both incoming and outgoing data streams are time aligned.

SDIN data stream must follow the AC'97 specification and be MSB justified with all non-valid bit positions (for assigned and/or unassigned time slots) stuffed with zeros. SDIN data is sampled by the ICH on the falling edge of BIT\_CLK.

### Input Slot 0: Tag Phase

Input slot 0 consists of a codec ready bit (bit 15), and slot valid bits for each subsequent slot in the frame (bits [14:3]).

The codec ready bit within slot 0 (bit 15) indicates whether the codec on the AC-link is ready for operation. If the codec ready bit in slot 0 is a zero, the codec is not ready for normal operation. When the AC-link codec ready bit is a 1, it indicates that the AC-link and codec control and status registers are in a fully operational state. The codec ready bits are visible through the Global Status register of the ICH. Software must further probe the Powerdown Control/Status register in the codec to determine exactly which subsections, if any, are ready.

Bits [14:3] in slot 0 indicate which slots of the input stream to the ICH contain valid data, just as in the output frame. The remaining bits in this slot are stuffed with zeros.

### Input Slot 1: Status Address Port / Slot Request Bits

The status port is used to monitor status of codec functions including, but not limited to, mixer settings and power management.

Slot 1 must echo the control register index, for historical reference, for the data to be returned in slot 2, assuming that slots 1 and 2 had been tagged valid by the codec in slot 0.

For multiple sample rate output, the codec examines its sample rate control registers, the state of its FIFOs, and the incoming SDOUT tag bits at the beginning of each audio output frame to determine which SLOTREQ bits to set active (low). SLOTREQ bits asserted during the current audio input frame signal which output slots require data from the controller in the next audio output frame. For fixed 48 kHz operation the SLOTREQ bits are always set active (low) and a sample is transferred each frame.

For multiple sample rate input, the tag bit for each input slot indicates whether valid data is present or not.

**Table 5-83. Input Slot 1 Bit Definitions**

| Bit   | Description   |
|-------|---|
| 19    | Reserved (Set to zero)  |
| 18:12 | Control Register Index (Stuffed with zeros if tagged invalid) |
| 11    | Slot 3 Request: PCM Left Channel*                             |
| 10    | Slot 4 Request: PCM Right Channel*                            |
| 9     | Slot 5 Request: Modem Line 1                                  |
| 8:2   | Slot Request 6-12: Not Implemented                            |
| 1:0   | Reserved (Stuffed with zeros)                                 |

**NOTE:** \*Slot 3 Request and Slot 4 Request bits must be the same value (i.e., set or cleared in tandem).

As shown in Table 5-83, slot 1 delivers codec control register read address and multiple sample rate slot request flags for all output slots of the controller. When a slot request bit is set by the codec, the controller will return data in that slot in the next output frame. Slot request bits for slots 3 and 4 are always set or cleared in tandem (i.e., both are set or cleared).

When set, the input slot 1 tag bit only pertains to Status Address Port data from a previous read. SLOTREQ bits are always valid independent of the slot 1 tag bit.

#### **Input Slot 2: Status Data Port**

The status data port receives 16-bit control register read data.

Bit [19:4]: Control Register Read Data

Bit [3:0]: Reserved.

#### **Input Slot 3: PCM Record Left Channel**

Input slot 3 is the left channel input of the codec. The ICH supports 16 bit sample resolution. Samples transmitted to the ICH must be in left/right channel order.

#### **Input Slot 4: PCM Record Right Channel**

Input slot 4 is the right channel input of the codec. The ICH supports 16 bit sample resolution. Samples transmitted to the ICH must be in left/right channel order.

#### **Input Slot 5: Modem Line**

Input slot 5 contains MSB justified modem data. The ICH supports 16 bit sample resolution.

#### **Input Slot 6: Optional Dedicated Microphone Record Data**

Input slot 6 is a third PCM system input channel available for dedicated use by a microphone. This input channel supplements a true stereo output which enables more precise echo cancellation algorithm for speakerphone applications. The ICH supports 16 bit resolution for slot 6 input.

#### **Input Slots 7-11: Reserved**

Input frame slots 7-11 are reserved for future use and should be stuffed with zeros by the codec, per the AC'97 specification.

#### **Input Slot 12: I/O status**

The status of the GPIOs configured as inputs are to be returned on this slot in every frame. The data returned on the latest frame is accessible to software by reading the register at offset 54h/D4h in the codec I/O space. Only the 16 MSBs are used to return GPI status. Bit 0 of this slot indicates the GPI status. Whenever a GPI changes state, this bit gets set for one frame by the codec. This bit can cause an interrupt to the processor if enabled via the Global Control register.

Reads from 54h/D4h will not be transmitted across the link in slot 1 and 2. The data from the most recent slot 12 is returned on reads from offset 54h/D4h.

## 5.17.4 Register Access

In the ICH implementation of the AC-link, up to two codecs can be connected to the SDOOUT pin. The following mechanism is used to address the primary and secondary codecs individually.

The primary device uses bit 19 of slot 1 as the direction bit to specify read or write. Bits [18:12] of slot 1 are used for the register index. To access the primary codec, the valid bits [14:13] for slots 1 and 2 must be set in slot 0, as shown in Table 5-84. Slot 1 is used to transmit the register address, and slot 2 is used to transmit data.

The secondary codec registers are accessed using slots 1 and 2 as described above, however the slot valid bits for slots 1 and 2 are marked invalid in slot 0 and the codec ID bit 0 (bit 0 of slot 0) is set to 1. This allows the secondary codec to monitor the slot valid bits of slots 1 and 2, and bit 0 of slot 0 to determine if the access is directed to the secondary codec. If the register access is targeted to the secondary codec, slot 1 and 2 will contain the address and data for the register access. Since slots 1 and 2 are marked invalid, the primary codec will ignore these accesses.

**Table 5-84. Output Tag Slot 0**

| Bit  | Primary Access Example | Secondary Access Example | Description   |
|------|------------------------|--------------------------|---|
| 15   | 1                      | 1                        | Frame Valid   |
| 14   | 1                      | 0                        | Slot 1 Valid, Command Address bit (Primary codec only)    |
| 13   | 1                      | 0                        | Slot 2 Valid, Command Data bit (Primary codec only)       |
| 12:3 | X                      | X                        | Slot 3-12 Valid   |
| 2    | 0                      | 0                        | Reserved  |
| 1:0  | 00                     | 01                       | Codec ID (00 reserved for primary; 01 indicate secondary) |

When accessing the codec registers, only one I/O cycle can be pending across the AC-link at any time. The ICH implements write posting on I/O writes across the AC-link (i.e., writes across the link are indicated as complete before they are actually sent across the link). In order to prevent a second I/O write from occurring before the first one is complete, software must monitor the CAS bit in the Codec Access Semaphore register which indicates that a codec access is pending. Once the CAS bit is cleared, then another codec access (read or write) can go through. The exception to this being reads to offset 54h/D4h (slot 12) which are returned immediately with the most recently received slot 12 data. Writes to offset 54h and D4h (primary and secondary codecs), get transmitted across the AC-link in slots 1 and 2 as a normal register access. Slot 12 is also updated immediately to reflect the data being written.

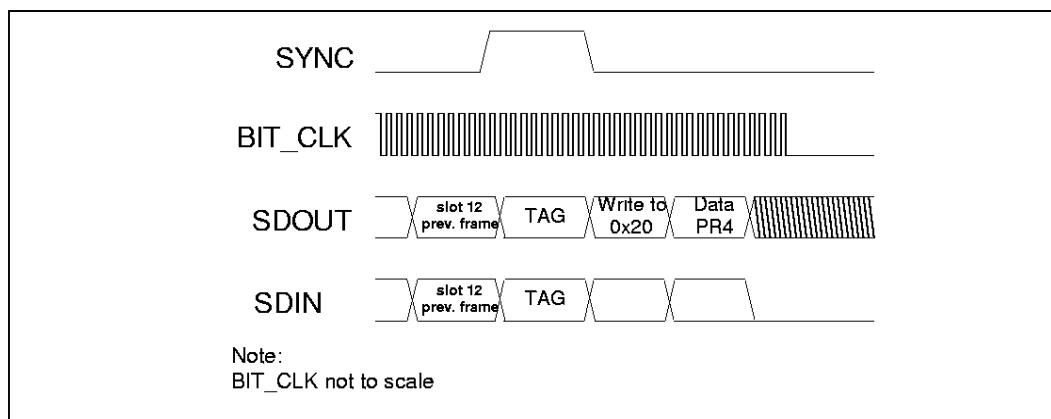
The controller will not issue back to back reads. It must get a response to the first read before issuing a second. In addition, codec reads and writes are only executed once across the link, and are not repeated.



## 5.18 AC-Link Low Power Mode

The AC-link signals can be placed in a low power mode. When the AC'97 Powerdown Register (26h) is programmed to the appropriate value, both BIT\_CLK and SDIN will be brought to, and held at a logic low voltage level.

Figure 5-20. AC-link Powerdown Timing



BIT\_CLK and SDIN transition low immediately following a write to the Powerdown Register (26h) with PR4. When the AC'97 controller driver is at the point where it is ready to program the AC-link into its low power mode, slots 1 and 2 are assumed to be the only valid stream in the audio output frame.

The AC'97 controller also drives AC\_SYNC, and SDOUT low after programming AC'97 to this low power, halted mode

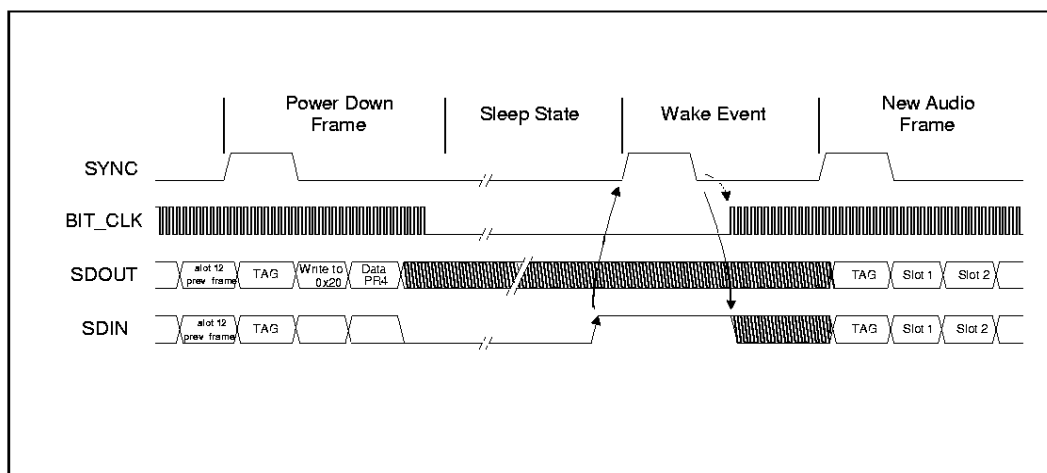
Once the codec has been instructed to halt BIT\_CLK, a special wake up protocol must be used to bring the AC-link to the active mode since normal output and input frames can not be communicated in the absence of BIT\_CLK. Once in a low power mode, the ICH provides three methods for waking up the AC-link; external wake event, cold reset and warm reset

**Note:** Before entering any low power mode where the link interface to the codec is expected to be powered down while the rest of the system is awake, the software must set the "Shut Off" bit in the control register.

### 5.18.1 External Wake Event

Codecs can signal the controller to wake the AC-link, and wake the system using SDIN.

Figure 5-21. SDIN Wake Signaling



The minimum SDIN wake up pulse width is 1 us. The rising edge of SDIN[0] or SDIN[1] causes the ICH to sequence through an AC-link warm reset and set the AC97\_STS bit in the GPE0\_STS register to wake the system. The primary codec must wait to sample AC\_SYNC high and low before restarting BIT\_CLK as diagrammed in Figure 5-21. The codec that signaled the wake event must keep its SDIN high until it has sampled AC\_SYNC having gone high, and then low.

The AC-link protocol provides for a cold reset and a warm reset. The type of reset used depends on the system's current power down state. Unless a cold or register reset (a write to the Reset register in the codec) is performed, wherein the AC'97 codec registers are initialized to their default values, registers are required to keep state during all power down modes.

Once powered down, activation of the AC-link via re-assertion of the AC\_SYNC signal must not occur for a minimum of 4 audio frame times following the frame in which the power down was triggered. When AC-link powers up, it indicates readiness via the codec ready bit.

### 5.18.2 AC'97 Cold Reset

A cold reset is achieved by asserting AC\_RST# for 1 us. By driving AC\_RST# low, BIT\_CLK, and SDOUT will be activated and all codec registers will be initialized to their default power on reset values. AC\_RST# is an asynchronous AC'97 input to the codec.

### 5.18.3 AC'97 Warm Reset

A warm reset will re-activate the AC-link without altering the current codec register values. A warm reset is signaled by driving AC\_SYNC high for a minimum of 1us in the absence of BIT\_CLK.

Within normal frames, AC\_SYNC is a synchronous AC'97 input to the codec. However, in the absence of BIT\_CLK, AC\_SYNC is treated as an asynchronous input to the codec used in the generation of a warm reset.

The codec must not respond with the activation of BIT\_CLK until AC\_SYNC has been sampled low again by the codec. This will prevent the false detection of a new frame.

**Note:** On receipt of wake up signalling from the codec, the digital controller will issue an interrupt if enabled. Software will then have to issue a warm or cold reset to the codec by setting the appropriate bit in the Global Control Register.

## 5.18.4 System Reset

Table 5-85 indicates the states of the link during various system reset and sleep conditions.

**Table 5-85. AC-link state during PCIRST#**

| Signal    | Power Plane         | I/O    | During PCIRST#  | After PCIRST# | S1                  | S3                 | S4/S5              |
|-----------|---------------------|--------|-----------------|---------------|---------------------|--------------------|--------------------|
| AC_RST#   | Resume <sup>3</sup> | Output | Low             | Low           | Cold Reset bit (Hi) | Low                | Low                |
| AC_SDOOUT | Core <sup>1</sup>   | Output | Low             | Running       | Low                 | Low                | Low                |
| AC_SYNC   | Core <sup>1</sup>   | Output | Low             | Running       | Low                 | Low                | Low                |
| BIT_CLK   | Core                | Input  | Driven by codec | Running       | Low <sup>2</sup>    | Low <sup>2</sup>   | Low <sup>2</sup>   |
| SDIN[1:0] | Resume              | Input  | Driven by codec | Running       | Low <sup>2,4</sup>  | Low <sup>2,4</sup> | Low <sup>2,4</sup> |

**NOTE:**

1. ICH core well outputs are used as strapping options for the ICH, sampled during system reset. These signals may have weak pullups/pulldowns on them. The ICH outputs will be driven to the appropriate level prior to AC\_RST# being deasserted, preventing a codec from entering test mode. Straps are tied to the core well to prevent leakage during a suspend state.
2. The pull-down resistors on these signals are only enabled when the AC-Link Shut Off bit in the AC'97 Global Control Register is set to 1. All other times, the pull-down resistor is disabled.
3. AC\_RST# will be held low during S3-S5. It cannot be programmed high during a suspend state.
4. SDIN[1:0] are driven low by codecs during normal states. If the codec is powered in suspend states it will hold SDIN[1:0] low. However, if the codec is not present or not powered in suspend, external pull-downs are required.

The transition of AC\_RST# to the deasserted state will only occur under driver control. In the S1 sleep state, the state of the AC\_RST# signal is controlled by the AC'97 Cold Reset# bit (bit 1) in the Global Control register. AC\_RST# will be asserted (low) by the ICH under the following conditions:

- RSMRST# (system reset, including the a reset of the resume well and PCIRST#)
- Mechanical power up (causes PCIRST#)
- Write to CF9h hard reset (causes PCIRST#)
- Transition to S3/S4/S5 sleep states (causes PCIRST#)
- Write to AC'97 Cold Reset# bit in the Global Control Register.

Hardware will never deassert AC\_RST# (i.e., never deasserts the Cold Reset# bit) automatically. Only software can deassert the Cold Reset# bit, and hence the AC\_RST# signal. This bit, while it resides in the core well, will remain cleared upon return from S3/S4/S5 sleep states. The AC\_RST# pin will remain actively driven from the resume well as indicated.



# Register and Memory Mapping

# 6

The ICH contains registers that are located in the processor's I/O space and memory space and sets of PCI configuration registers that are located in PCI configuration space. This chapter describes the ICH I/O and memory maps at the register-set level. Register access is also described. Register-level address maps and Individual register bit descriptions are provided in the following chapters. The following notations and definitions are used in the register/instruction description chapters.

|         |  |
|---------|--|
| RO      | Read Only. In some cases, If a register is read only, writes to this register location have no effect. However, in other cases, two separate registers are located at the same location where a read accesses one of the registers and a write accesses the other register. See the I/O and memory map tables for details.   |
| WO      | Write Only. In some cases, If a register is write only, reads to this register location have no effect. However, in other cases, two separate registers are located at the same location where a read accesses one of the registers and a write accesses the other register. See the I/O and memory map tables for details.  |
| R/W     | Read/Write. A register with this attribute can be read and written.  |
| R/WC    | Read/Write Clear. A register bit with this attribute can be read and written. However, a write of 1 clears (sets to 0) the corresponding bit and a write of 0 has no effect.   |
| Default | When ICH is reset, it sets its registers to predetermined default states. The default state represents the minimum functionality feature set required to successfully bring up the system. Hence, it does not represent the optimal system configuration. It is the responsibility of the system initialization software to determine configuration, operating parameters, and optional system features that are applicable, and to program the ICH registers accordingly. |
| Bold    | Register bits that are highlighted in bold text indicate that the bit is implemented in the ICH. Register bits that are not implemented or are harwired will remain in plain text.   |

## 6.1 PCI Devices and Functions

The ICH incorporates a variety of PCI functions as shown in Table 6-1. These functions are divided into two logical devices (30, and 31). Device 30 is the Hub Interface-To-PCI bridge. Device 31 contains all the other PCI functions.

If for some reason, the particular system platform does not want to support any one of the Functions 1-6, they can individually be disabled. When a function is disabled, it does not appear at all to the software. A disabled function will not respond to any register reads or writes. This is intended to prevent software from thinking that a function is present (and reporting it to the end-user).

**Table 6-1. PCI Devices and Functions**

| Device:Function      | Function Description   |
|----------------------|--|
| Device 30:Function 0 | Hub Interface to PCI Bridge  |
| Device 31:Function 0 | PCI to LPC Bridge<br>(includes: DMA, Timers, compatible interrupt controller, APIC, RTC, processor interface control, power management control, System Management control, and GPIO control) |
| Device 31:Function 1 | IDE Controller   |
| Device 31:Function 2 | USB Controller   |
| Device 31:Function 3 | SMBus Controller   |
| Device 31:Function 4 | Reserved   |
| Device 31:Function 5 | AC'97 Audio Controller   |
| Device 31:Function 6 | AC'97 Modem Controller   |

### 6.1.1 PCI Configuration Map

Each PCI function on the ICH has a set of PCI configuration registers. The register address map tables for these register sets are included at the beginning of the chapter for the particular function. Refer to Table A-1 for a complete list of all PCI Configuration Registers.

Configuration Space registers are accessed through configuration cycles on the PCI bus by the Host bridge using configuration mechanism #1 detailed in the PCI 2.1 specification.

Some of the PCI registers contain reserved bits. Software must deal correctly with fields that are reserved. On reads, software must use appropriate masks to extract the defined bits and not rely on reserved bits being any particular value. On writes, software must ensure that the values of reserved bit positions are preserved. That is, the values of reserved bit positions must first be read, merged with the new values for other bit positions and then written back. Note the software does not need to perform read, merge, write operation for the configuration address register.

In addition to reserved bits within a register, the configuration space contains reserved locations. Software should not write to reserved PCI configuration locations in the device-specific region (above address offset 3Fh).

## 6.1.2 Standard PCI Bus Configuration Mechanism

The PCI Bus defines a slot based “configuration space” that allows each device to contain up to 8 functions with each function containing up to 256 8-bit configuration registers. The PCI specification defines two bus cycles to access the PCI configuration space: Configuration Read and Configuration Write. Memory and I/O spaces are supported directly by the processor. Configuration space is supported by a mapping mechanism implemented within the ICH. The PCI specification defines two mechanisms to access configuration space, Mechanism #1 and Mechanism #2. The ICH only supports Mechanism #1.

Configuration cycles for PCI Bus #0 devices #2 through #31, and for PCI Bus numbers greater than 0 will be sent towards the ICH from the host controller. The ICH compares the non-zero Bus Number with the Secondary Bus Number and Subordinate Bus number registers of its P2P bridge to determine if the configuration cycle is meant for Primary PCI or a downstream PCI bus.

**Type 0 to Type 0 Forwarding:** When a Type 0 configuration cycle is received on hub link, the ICH forwards these cycles to PCI and then reclaims them. The ICH uses address bits AD[15:14] to communicate the ICH device numbers in Type 0 configuration cycles. If the Type 0 cycle on hub link specifies any device number other than 30 or 31, the ICH will not set any address bits in the range AD[31:11] during the corresponding transaction on PCI. Table 6-2 shows the device number translation.

**Table 6-2. Device Number Translation**

| Device # In Hub link Type 0 Cycle | AD[31:11] During Address Phase of Type 0 Cycle on PCI |
|-----------------------------------|---|
| 0 through 29                      | 0000000000000000_00000b                               |
| 30                                | 0000000000000000_01000b                               |
| 31                                | 0000000000000000_10000b                               |

The ICH logic will generate single D-word configuration read and write cycles on the PCI bus. The ICH will generate a Type 0 configuration cycle for configurations to the bus number matching the PCI bus. Type 1 configuration cycles will be converted to Type 0 cycles in this case. If the cycle is targeting a device behind an external bridge, the ICH will run a Type 1 cycle on the PCI bus.

**Type 1 to Type 0 Conversion:** When the bus number for the Type 1 configuration cycle matches the PCI (Secondary) bus number, the ICH will convert the address as follows:

1. For device numbers 0 through 15, only one bit of the PCI address [31:16] will be set. If the device number is 0, AD[16] is set; if the device number is 1, AD[17] is set; etc.
2. The ICH will always drive 0's on bits AD[15:11] when converting Type 1 configurations cycles to Type 0 configuration cycles on PCI.
3. Address bits [10:1] will also be passed unchanged to PCI.
4. Address bit [0] will be changed to '0'.

## 6.2 I/O Map

The I/O map is divided into Fixed and Variable address ranges. Fixed ranges cannot be moved, but in some cases can be disabled. Variable ranges can be moved and can also be disabled.

### 6.2.1 Fixed I/O Address Ranges

Table 6-3 shows the Fixed I/O decode ranges from the processor perspective. Note that for each I/O range, there may be separate behavior for reads and writes. The hub interface cycles that go to target ranges that are marked as “Reserved” will not be decoded by the ICH, and will be passed to PCI. If a PCI master targets one of the fixed I/O target ranges, it will be positively decoded by the ICH in Medium speed.

Refer to Table A-2 for a complete list of all fixed I/O registers. Address ranges that are not listed or marked “Reserved” are NOT decoded by the ICH (unless assigned to one of the variable ranges).

**Table 6-3. Fixed I/O Ranges Decoded by ICH (Sheet 1 of 2)**

| I/O Address | Read Target          | Write Target           | Internal Unit       |
|-------------|----------------------|------------------------|---------------------|
| 00h–08h     | DMA Controller       | DMA Controller         | DMA                 |
| 09h–0Eh     | Reserved             | DMA Controller         | DMA                 |
| 0Fh         | DMA Controller       | DMA Controller         | DMA                 |
| 10h–18h     | DMA Controller       | DMA Controller         | DMA                 |
| 19h–1Eh     | Reserved             | DMA Controller         | DMA                 |
| 1Fh         | DMA Controller       | DMA Controller         | DMA                 |
| 20h–21h     | Interrupt Controller | Interrupt Controller   | Interrupt           |
| 24h–25h     | Interrupt Controller | Interrupt Controller   | Interrupt           |
| 28h–29h     | Interrupt Controller | Interrupt Controller   | Interrupt           |
| 2Ch–2Dh     | Interrupt Controller | Interrupt Controller   | Interrupt           |
| 30h–31h     | Interrupt Controller | Interrupt Controller   | Interrupt           |
| 34h–35h     | Interrupt Controller | Interrupt Controller   | Interrupt           |
| 38h–39h     | Interrupt Controller | Interrupt Controller   | Interrupt           |
| 3Ch–3Dh     | Interrupt Controller | Interrupt Controller   | Interrupt           |
| 40h–42h     | Timer/Counter        | Timer/Counter          | PIT (8254)          |
| 43h         | Reserved             | Timer/Counter          | PIT                 |
| 50h–52h     | Timer/Counter        | Timer/Counter          | PIT                 |
| 53h         | Reserved             | Timer/Counter          | PIT                 |
| 61h         | NMI Controller       | NMI Controller         | Processor Interface |
| 70h         | Reserved             | NMI and RTC Controller | RTC                 |
| 71h         | RTC Controller       | RTC Controller         | RTC                 |
| 72h         | RTC Controller       | NMI and RTC Controller | RTC                 |
| 73h         | RTC Controller       | RTC Controller         | RTC                 |
| 74h         | RTC Controller       | NMI and RTC Controller | RTC                 |
| 75h         | RTC Controller       | RTC Controller         | RTC                 |
| 76h         | RTC Controller       | NMI and RTC Controller | RTC                 |



Table 6-3. Fixed I/O Ranges Decoded by ICH (Sheet 2 of 2)

| I/O Address | Read Target                 | Write Target                       | Internal Unit       |
|-------------|-----------------------------|------------------------------------|---------------------|
| 77h         | RTC Controller              | RTC Controller                     | RTC                 |
| 80h         | DMA Controller              | DMA Controller and LPC or PCI      | DMA                 |
| 81h–83h     | DMA Controller              | DMA Controller                     | DMA                 |
| 84h–86h     | DMA Controller              | DMA Controller and LPC or PCI      | DMA                 |
| 87h         | DMA Controller              | DMA Controller                     | DMA                 |
| 88h         | DMA Controller              | DMA Controller and LPC or PCI      | DMA                 |
| 89h–8Bh     | DMA Controller              | DMA Controller                     | DMA                 |
| 8Ch–8Eh     | DMA Controller              | DMA Controller and LPC or PCI      | DMA                 |
| 08Fh        | DMA Controller              | DMA Controller                     | DMA                 |
| 92h         | Reset Generator             | Reset Generator                    | Processor Interface |
| A0h–A1h     | Interrupt Controller        | Interrupt Controller               | Interrupt           |
| A4h–A5h     | Interrupt Controller        | Interrupt Controller               | Interrupt           |
| A8h–A9h     | Interrupt Controller        | Interrupt Controller               | Interrupt           |
| ACH–ADh     | Interrupt Controller        | Interrupt Controller               | Interrupt           |
| B0h–B1h     | Interrupt Controller        | Interrupt Controller               | Interrupt           |
| B2h–B3h     | Power Management            | Power Management                   | TBD                 |
| B4h–B5h     | Interrupt Controller        | Interrupt Controller               | Interrupt           |
| B8h–B9h     | Interrupt Controller        | Interrupt Controller               | Interrupt           |
| BCh–BDh     | Interrupt Controller        | Interrupt Controller               | Interrupt           |
| C0h–D1h     | DMA Controller              | DMA Controller                     | DMA                 |
| D2h–DDh     | Reserved                    | DMA Controller                     | DMA                 |
| DEh–DFh     | DMA Controller              | DMA Controller                     | DMA                 |
| F0h         | See Note 3                  | FERR#/IGNNE#/ Interrupt Controller | Processor Interface |
| 170h–177h   | IDE Controller <sup>1</sup> | IDE Controller <sup>1</sup>        | IDE                 |
| 1F0h–1F7h   | IDE Controller <sup>2</sup> | IDE Controller <sup>2</sup>        | IDE                 |
| 376h        | IDE Controller <sup>1</sup> | IDE Controller <sup>1</sup>        | IDE                 |
| 3F6h        | IDE Controller <sup>2</sup> | IDE Controller <sup>2</sup>        | IDE                 |
| 4D0h–4D1h   | Interrupt Controller        | Interrupt Controller               | IDE                 |
| CF9h        | Reset Generator             | Reset Generator                    | Processor Interface |

**NOTES:**

1. Only if IDE Standard I/O space is enabled for Primary Drive. Otherwise, the target is PCI.
2. Only if IDE Standard I/O space is enabled for Secondary Drive. Otherwise, the target is PCI.
3. If POS\_DEC\_EN bit is enabled, reads from F0h will not be decoded by the ICH. If POS\_DEC\_EN is not enabled, reads from F0h will forward to LPC.

## 6.2.2 Variable I/O Decode Ranges

Table 6-4 shows the Variable I/O Decode Ranges. They are set using Base Address Registers (BARs) or other configuration bits in the various PCI configuration spaces. The PNP software (PCI or ACPI) can use their configuration mechanisms to set and adjust these values.

When a cycle is detected on hub interface, the ICH will positively decode the cycle. If the response is on the behalf of an LPC device, ICH will forward the cycle to the LPC I/F.

Refer to Table A-3 for a complete list of all variable I/O registers.

**Warning:** The Variable I/O Ranges should not be set to conflict with the Fixed I/O Ranges. Unpredictable results if the configuration software allows conflicts to occur. The ICH does not perform any checks for conflicts.

**Table 6-4. Variable I/O Decode Ranges**

| Range Name             | Mappable                  | Size (Bytes)          | Target           |
|------------------------|---------------------------|-----------------------|------------------|
| ACPI                   | Anywhere in 64K I/O Space | 64                    | Power Management |
| IDE                    | Anywhere in 64K I/O Space | 16                    | IDE Unit         |
| USB                    | Anywhere in 64K I/O Space | 32                    | USB Unit         |
| SMBus                  | Anywhere in 64K I/O Space | 16                    | SMB Unit         |
| AC'97 Audio Mixer      | Anywhere in 64K I/O Space | 256                   | AC'97 Unit       |
| AC'97 Bus Master       | Anywhere in 64K I/O Space | 64                    | AC'97 Unit       |
| AC'97 Modem Mixer      | Anywhere in 64K I/O Space | 256                   | AC'97 Unit       |
| TCO                    | 96 Bytes above ACPI Base  | 32                    | TCO Unit         |
| GPIO                   | Anywhere in 64K I/O Space | 64                    | GPIO Unit        |
| Parallel Port          | 3 ranges in 64K I/O Space | 8                     | LPC Peripheral   |
| Serial Port 1          | 8 Ranges in 64K I/O Space | 8                     | LPC Peripheral   |
| Serial Port 2          | 8 Ranges in 64K I/O Space | 8                     | LPC Peripheral   |
| Floppy Disk Controller | 2 Ranges in 64K I/O Space | 8                     | LPC Peripheral   |
| MIDI                   | 4 Ranges in 64K I/O Space | 2                     | LPC Peripheral   |
| MSS                    | 4 Ranges in 64K I/O Space | 8                     | LPC Peripheral   |
| SoundBlaster           | 2 Ranges in 64K I/O Space | 32                    | LPC Peripheral   |
| AdLib                  | 2 Ranges in 64K I/O Space | 2                     | LPC Peripheral   |
| LPC Generic 1          | Anywhere in 64K I/O Space | 128 bytes (with mask) | LPC Peripheral   |
| LPC Generic 2          | Anywhere in 64K I/O Space | 16 bytes (with mask)  | LPC Peripheral   |

## 6.3 Memory Map

Table 6-5 shows (from the processor perspective) the memory ranges that the ICH will decode. Cycles that arrive from the host controller will first be driven out on PCI. The ICH may then claim the cycle for it to be forwarded to LPC or claimed by the internal APIC. If subtractive decode is enabled, the cycle can be forwarded to LPC.

PCI cycles generated by an external PCI master will be positively decoded unless it falls in the PCI-PCI bridge forwarding range (those addresses are reserved for PCI peer-to-peer traffic). If the cycle is not in the I/O APIC or LPC ranges, it will be forwarded up the hub interface to the host controller.

**Table 6-5. Memory Decode Ranges from Processor Perspective**

| Memory Range  | Target  |
|---|---|
| 00000000h–000DFFFFh<br>00100000h –TOM (Top of Memory) | Main Memory                                     |
| 000E0000h–000FFFFFFh                                  | FWH   |
| FEC00000h–FEC00100h                                   | I/O APIC inside ICH                             |
| FF800000h–FFF80000h                                   | FWH<br>Refer to Section 8.1.3 for more details. |
| FFF80000h–FFFFFFFFh                                   | FWH   |
| All Other   | PCI   |



# Hub Interface-to-PCI Bridge Registers (D30:F0)

# 7

The hub interface to PCI Bridge resides in PCI Device 30, Function 0 on bus #0. This portion of the ICH implements the buffering and control logic between PCI and the hub interface. The arbitration for the PCI bus is handled by this PCI device. The PCI decoder in this device must decode the ranges for the hub interface. All register contents will be lost when core well power is removed.

## 7.1 PCI Configuration Registers (D30:F0)

**Note:** Reserved registers are read only and are not shown. Reads to unlisted reserved registers should return all 0's

Table 7-1. PCI Configuration Map (HUB-PCI—D30:F0) (Sheet 1 of 2)

| Offset | Mnemonic      | Register Name/Function         | Default   | Type |
|--------|---------------|--------------------------------|---|------|
| 00–01h | VID           | Vendor ID                      | 8086h   | RO   |
| 02–03h | DID           | Device ID                      | 2418h<br>(ICH: 82801AA)<br>2428h<br>(ICH0: 82801AB) | RO   |
| 04–05h | CMD           | PCI Device Command Register    | 0001h   | R/W  |
| 06–07h | PD_STS        | PCI Device Status Register     | 0080h   | R/W  |
| 08h    | RID           | Revision ID                    | Note 1  | RO   |
| 0Ah    | SCC           | Sub Class Code                 | 04h   | RO   |
| 0Bh    | BCC           | Base Class Code                | 06h   | RO   |
| 0Dh    | PMLT          | Primary Master Latency Timer   | 00h   | RO   |
| 0Eh    | HEADTYP       | Header Type                    | 01h   | RO   |
| 18h    | PBUS_NUM      | Primary Bus Number             | 00h   | R/W  |
| 19h    | SBUS_NUM      | Secondary Bus Number           | 00h   | R/W  |
| 1Ah    | SUB_BUS_NUM   | Subordinate Bus Number         | 00h   | R/W  |
| 1Bh    | SMLT          | Secondary Master Latency Timer | 00h   | R/W  |
| 1Ch    | IOBASE        | IO Base Register               | F0h   | R/W  |
| 1Dh    | IOLIM         | IO Limit Register              | 00h   | R/W  |
| 1E–1Fh | SECSTS        | Secondary Status Register      | 0280h   | R/W  |
| 20–21h | MEMBASE       | Memory Base                    | FFF0h   | R/W  |
| 22–23h | MEMLIM        | Memory Limit                   | 0000h   | R/W  |
| 24–25h | PREF_MEM_BASE | Prefetchable Memory Base       | 0000h   | R/W  |
| 26–27h | PREF_MEM_MLT  | Prefetchable Memory Limit      | 0000h   | R/W  |
| 30–31h | IOBASE_HI     | I/O Base Upper 16 Bits         | 0000h   | RO   |
| 32–33h | IOLIMIT_HI    | I/O Limit Upper 16 Bits        | 0000h   | RO   |

Table 7-1. PCI Configuration Map (HUB-PCI—D30:F0) (Sheet 2 of 2)

| Offset | Mnemonic     | Register Name/Function     | Default | Type |
|--------|--------------|----------------------------|---------|------|
| 3Ch    | INT_LINE     | Interrupt Line             | 00h     | RO   |
| 3E–3Fh | BRIDGE_CNT   | Bridge Control             | 0000h   | R/W  |
| 50–51h | CNF          | ICH Configuration Register | 0000h   | R/W  |
| 70h    | MTT          | Multi-Transaction Timer    | 00h     | R/W  |
| 82h    | PCI_MAST_STS | PCI Master Status          | 00h     | R/W  |
| 90h    | ERR_CMD      | Error Command Register     | 00h     | R/W  |
| 92h    | ERR_STS      | Error Status Register      | 00h     | R/W  |

**NOTES:**

1. Refer to ICH Specification Updates for the value of the Revision ID.
2. Addresses that are not listed in the above table are reserved and read only.

### 7.1.1 VID—Vendor ID Register (HUB-PCI—D30:F0)

Offset Address: 00–01h                      Attribute: RO  
 Default Value: 8086h                      Size: 16 bits

| Bit  | Description   |
|------|---|
| 15:0 | <b>Vendor Identification Number.</b> This is a 16-bit value assigned to Intel. Intel VID = 8086h. |

### 7.1.2 DID—Device ID Register (HUB-PCI—D30:F0)

Offset Address: 02–03h                      Attribute: RO  
 Size: 16 bits  
 Default Value: 2418h (ICH: 82801AA)  
 2428h (ICH: 82801AB)

| Bit  | Description   |
|------|---|
| 15:0 | <b>Device Identification Number.</b> This is a 16 bit value assigned to the ICH hub interface to PCI bridge (i.e. Device #2). See default above for value in this register. |



### 7.1.4 PD\_STS—Primary Device Status Register (HUB-PCI—D30:F0)

Offset Address: 06–07h Attribute: R/W  
 Default Value: 0080h Size: 16 bits

For the writable bits in this register, writing a '1' will clear the bit. Writing a '0' to the bit will have no effect.

| Bit  | Description   |
|------|---|
| 15   | <p><b>ICH (82801AA):</b><br/> <b>Detected Parity Error (DPE) —R/WC.</b><br/>           1 = This bit indicates that the ICH detected a parity error on the hub interface 1. This bit gets set even if the Parity Error Response bit (bit 6 of offset 04) is not set.<br/>           0 = This bit is cleared by writing a '1' to this location.</p> <p><b>ICH0 (82801AB):</b><br/>           Detected Parity Error—RO. Hardwired to '0'</p>   |
| 14   | <p><b>ICH (82801AA):</b><br/> <b>Received System Error (SSE) — R/W.</b><br/>           1 = The ICH sets this bit when an address, or command parity error, or special cycles data parity error is detected on the PCI bus, and the Parity Error Response bit (D30:F0, Offset 04h, bit 6) is set.<br/>           Note that if this bit is set because of parity error and the D30:F0 SERR_EN bit (Offset 04h, bit 8) is also set, the ICH will generate an NMI (or SMI# if NMI routed to SMI#)<br/>           0 = This bit is cleared by software writing a 1.</p> <p><b>ICH0 (82801AB):</b><br/> <b>Received System Error (SSE) — R/W.</b><br/>           1 = The ICH0 sets this bit when a system error occurs. Refer to Section 5.1.4, "SERR#/PERR#/NMI# Functionality" on page 5-2 for additional information.<br/>           0 = This bit is cleared by software writing a 1.</p> |
| 13   | <p><b>Received Master Abort (RMA) —R/WC.</b><br/>           1 = ICH received a master abort from the hub interface device (host controller).<br/>           0 = This bit is cleared by software writing a 1.</p>  |
| 12   | <p><b>Received Target Abort (RTA) —R/WC.</b><br/>           1 = ICH received a target abort from the hub interface device (host controller). The TCO logic can cause an SMI#, NMI, or interrupt based on this bit getting set.<br/>           0 = This bit is cleared by software writing a 1.</p>  |
| 11   | <p><b>Signaled Target Abort (STA) —R/W.</b><br/>           1 = ICH signals a target abort condition on the hub interface. Note: an implementation simplification is to set this bit along with the PCI (secondary) bus Received Target Abort bit during a "completion required" access.<br/>           0 = This bit is cleared by software writing a 1.</p>   |
| 10:9 | <p>DEVSEL# Timing Status—RO.<br/>           00h = Fast timing. This register applies to the hub interface; therefore, this field does not matter.</p>   |
| 8    | <p><b>ICH (82801AA):</b><br/> <b>Data Parity Error Detected (DPD) —R/W.</b> With the PERR signal removed from the hub interface, the ICH must interpret this bit differently than it is in the PCI spec.<br/>           1 = ICH (82801AA) detects a parity error on the hub interface and the Parity Error Response bit in the Command Register (offset 04h, bit 6) is set.<br/>           0 = This bit is cleared by software writing a 1.</p> <p><b>ICH0 (82801AB):</b><br/>           Data Parity Error Detected—RO. Hardwired to '0'.</p>   |



| Bit | Description  |
|-----|--|
| 7   | Fast Back to Back—RO. This bit is hardwired to '1'.  |
| 6   | User Definable Features (UDF) —RO. Hardwired to '0'. |
| 5   | 66 MHz Capable—RO. Hardwired to '0'.                 |
| 4:0 | Reserved.  |

### 7.1.5 RID—Revision ID Register (HUB-PCI—D30:F0)

Offset Address: 08h                                   Attribute: RO  
 Default Value: See ICH Spec. Updates        Size: 8 bits

| Bit | Description   |
|-----|---|
| 7:0 | <b>Revision Identification Number.</b> 8-bit value that indicates the revision number for the ICH hub interface to PCI bridge. Refer to ICH Specification Updates for this value. |

### 7.1.6 SCC—Sub-Class Code Register (HUB-PCI—D30:F0)

Offset Address: 0Ah                                   Attribute: RO  
 Default Value: 04h                                Size: 8 bits

| Bit | Description   |
|-----|---|
| 7:0 | <b>Sub-Class Code.</b> 8-bit value that indicates the category of bridge for the ICH hub interface to PCI bridge. The code is 04h indicating a PCI-to-PCI bridge. |

### 7.1.7 BCC—Base-Class Code Register (HUB-PCI—D30:F0)

Offset Address: 0Bh                                   Attribute: RO  
 Default Value: 06h                                Size: 8 bits

| Bit | Description  |
|-----|--|
| 7:0 | <b>Base Class Code.</b> 8-bit value that indicates the type of device for the ICH hub interface to PCI bridge. The code is 06h indicating a bridge device. |

### 7.1.8 PMLT—Primary Master Latency Timer Register (HUB-PCI—D30:F0)

Offset Address: 0Dh                                   Attribute: RO  
 Default Value: 00h                                Size: 8 bits

This register does not apply to hub interface.

| Bit | Description                            |
|-----|--|
| 7:3 | Master Latency Count. Not implemented. |
| 2:0 | Reserved.                              |

### 7.1.9 HEADTYP—Header Type Register (HUB-PCI—D30:F0)

Offset Address: 0Eh                      Attribute: RO  
 Default Value: 01h                      Size: 8 bits

| Bit | Description   |
|-----|---|
| 7   | <b>Multi-function Device.</b> This bit is '0' to indicate a single function device.   |
| 6:0 | <b>Header Type.</b> 8-bit field identifies the header layout of the configuration space, which is a PCI-to-PCI bridge in this case. |

### 7.1.10 PBUS\_NUM—Primary Bus Number Register (HUB-PCI—D30:F0)

Offset Address: 18h                      Attribute: R/W  
 Default Value: 00h                      Size: 8 bits

| Bit | Description  |
|-----|--|
| 7:0 | <b>Primary Bus Number.</b> This field indicates the bus number of the hub interface and is hardwired to 00h. |

### 7.1.11 SBUS\_NUM—Secondary Bus Number Register (HUB-PCI—D30:F0)

Offset Address: 19h                      Attribute: R/W  
 Default Value: 00h                      Size: 8 bits

| Bit | Description  |
|-----|--|
| 7:0 | <b>Secondary Bus Number.</b> This field indicates the bus number of PCI. Note: when this number is equal to the primary bus number (i.e. bus #0), the ICH will run hub interface configuration cycles to this bus number as Type 1 configs on PCI. |

### 7.1.12 SUB\_BUS\_NUM—Subordinate Bus Number Register (HUB-PCI—D30:F0)

Offset Address: 1A                      Attribute: R/W  
 Default Value: 00h                      Size: 8 bits

| Bit | Description  |
|-----|--|
| 7:0 | <b>Subordinate Bus Number.</b> This field specifies the highest PCI bus number below the hub interface to PCI bridge. If a Type 1 configuration cycle from the hub interface does not fall in the Secondary-to-Subordinate Bus ranges of Device 30, the ICH must indicate a master abort back to the hub interface (with all 1's, in the case of reads). |

### 7.1.13 SMLT—Secondary Master Latency Timer Register (HUB-PCI—D30:F0)

Offset Address: 1Bh Attribute: R/W  
 Default Value: 00h Size: 8 bits

This Master Latency Timer (MLT) controls the amount of time that the ICH will continue to burst data as a master on the PCI bus. When the ICH starts the cycle after being granted the bus, the counter is loaded and starts counting down from the assertion of FRAME#. If the internal grant to this device is removed, then the expiration of the MLT counter will result in the deassertion of FRAME#. If the internal grant has not been removed, then the ICH can continue to own the bus.

| Bit | Description   |
|-----|---|
| 7:3 | <b>Master Latency Count.</b> 5-bit value that indicates the number of PCI clocks, in 8-clock increments, that the ICH will remain as master of the bus. |
| 2:0 | Reserved.   |

### 7.1.14 IOBASE—I/O Base Register (HUB-PCI—D30:F0)

Offset Address: 1Ch Attribute: R/W  
 Default Value: F0h Size: 8 bits

| Bit | Description   |
|-----|---|
| 7:4 | <b>I/O Address Base bits [15:12]—R/W.</b> I/O Base bits corresponding to address lines 15:12 for 4 KB alignment. Bits 11:0 are assumed to be padded to 000h.  |
| 3:0 | <b>I/O Addressing Capability—RO.</b> This is hardwired to 0h, indicating that the hub interface to PCI bridge does not support 32-bit I/O addressing. This means that the I/O base & limit upper address registers must be read only. |

### 7.1.15 IOLIM—I/O Limit Register (HUB-PCI—D30:F0)

Offset Address: 1Dh Attribute: R/W  
 Default Value: 00h Size: 8 bits

| Bit | Description   |
|-----|---|
| 7:4 | <b>I/O Address Limit bits [15:12]—R/W.</b> I/O Base bits corresponding to address lines 15:12 for 4 KB alignment. Bits 11:0 are assumed to be padded to FFFh.   |
| 3:0 | <b>I/O Addressing Capability—RO.</b> This is hardwired to 0h, indicating that the hub interface to PCI bridge does not support 32-bit I/O addressing. This means that the I/O base & limit upper address registers must be read only. |

## 7.1.16 SECSTS—Secondary Status Register (HUB-PCI—D30:F0)

Offset Address: 1E–1Fh                      Attribute: R/W  
 Default Value: 0280h                      Size: 16 bits

For the writable bits in this register, writing a ‘1’ will clear the bit. Writing a ‘0’ to the bit will have no effect.

| Bit  | Description  |
|------|--|
| 15   | <p><b>ICH (82801AA):</b><br/> <b>Detected Parity Error (DPE) —R/WC.</b><br/>           1 = ICH (82801AA) detected a parity error on the PCI bus.<br/>           0 = This bit is cleared by software writing a 1.</p> <p><b>ICH0 (82801AB):</b><br/>           Detected Parity Error—RO. Hardwired to ‘0’.</p>  |
| 14   | <p><b>Received System Error (SSE) —R/WC.</b><br/>           1 = SERR# assertion is received on PCI.<br/>           0 = This bit is cleared by software writing a 1.</p>  |
| 13   | <p><b>Received Master Abort (RMA) —R/WC.</b><br/>           1 = hub interface to PCI cycle is master-aborted on PCI.<br/>           0 = This bit is cleared by software writing a 1.</p>   |
| 12   | <p><b>Received Target Abort (RTA) —R/WC.</b><br/>           1 = hub interface to PCI cycle is target-aborted on PCI. For “completion required” cycles from the hub interface, this event should also set the Signaled Target Abort in the Primary Status Register in this device, and the ICH must send the “target abort” status back to the hub interface.<br/>           0 = This bit is cleared by software writing a 1.</p>   |
| 11   | <p>Signaled Target Abort (STA) —RO. The ICH does not generate target aborts and does not forward the target abort response from the hub interface to the PCI interface.</p>  |
| 10:9 | <p>DEVSEL# Timing Status—RO.<br/>           01h = Medium timing.</p>   |
| 8    | <p><b>ICH (82801AA):</b><br/> <b>Data Parity Error Detected—R/WC</b><br/>           1 = The ICH (82801AA) sets this bit when all of the following three conditions are met:<br/>           - The Parity Error Response Enable bit in the Bridge Control Register (bit 0, offset 3Eh) is set<br/>           - USB, AC’97 or IDE is a Master<br/>           - PERR# asserts during a write cycle <b>OR</b> a parity error is detected internally during a read cycle<br/>           0 = This bit is cleared by software writing a 1.</p> <p><b>ICH0 (82801AB):</b><br/>           Data Parity Error Detected—RO. Hardwired to ‘0’.</p> |
| 7    | <p><b>Fast Back to Back—RO.</b> Hardwired to ‘1’ to indicate that the PCI to hub interface target logic is capable of receiving fast back-to-back cycles.</p>  |
| 6    | <p>User Definable Features (UDF) —RO. Hardwired to ‘0’.</p>  |
| 5    | <p>66 MHz Capable—RO. Hardwired to ‘0’.</p>  |
| 4:0  | <p>Reserved.</p>   |



### 7.1.20 **PREF\_MEM\_MLT—Prefetchable Memory Limit Register (HUB-PCI—D30:F0)**

Offset Address: 26h–27h                      Attribute: R/W  
 Default Value: 00000000h                  Size: 16-bit

| Bit  | Description  |
|------|--|
| 15:4 | <b>Prefetchable Memory Address Limit—RW.</b> Defines the limit address of the prefetchable memory address range for PCI. These 12 bits correspond to address bits 31:20. |
| 3:0  | Reserved. RO   |

### 7.1.21 **IOBASE\_HI—I/O Base Upper 16 Bits Register (HUB-PCI—D30:F0)**

Offset Address: 30–31h                      Attribute: RO  
 Default Value: 0000h                      Size: 16 bits

| Bit  | Description  |
|------|--|
| 15:0 | I/O Address Base Upper 16 bits [31:16]. Not supported; hardwired to 0. |

### 7.1.22 **IOLIM\_HI—I/O Limit Upper 16 Bits Register (HUB-PCI—D30:F0)**

Offset Address: 32–33h                      Attribute: RO  
 Default Value: 0000h                      Size: 16 bits

| Bit  | Description   |
|------|---|
| 15:0 | I/O Address Limit Upper 16 bits [31:16]. Not supported; hardwired to 0. |

### 7.1.23 **INT\_LINE—Interrupt Line Register (HUB-PCI—D30:F0)**

Offset Address: 3Ch                          Attribute: RO  
 Default Value: 00h                          Size: 8 bits

| Bit | Description   |
|-----|---|
| 7:0 | Interrupt Line Routing. Hardwired to 00h. The bridge does not generate interrupts, and interrupts from downstream devices are routed around the bridge. |

### 7.1.24 BRIDGE\_CNT—Bridge Control Register (HUB-PCI—D30:F0)

Offset Address: 3E–3Fh Attribute: R/W  
 Default Value: 0000h Size: 16 bits

| Bit  | Description   |
|------|---|
| 15:8 | Reserved.   |
| 7    | Fast Back to Back Enable—RO. Hardwired to '0'. The PCI logic will not generate fast back-to-back cycles on the PCI bus.   |
| 6    | Secondary Bus Reset—RO. hardwired to '0'. The ICH does not follow the P2P bridge reset scheme; Software-controlled resets are implemented in the PCI-LPC device.  |
| 5    | <b>Master Abort Mode—R/W.</b> The ICH ignores this bit. However, this bit is read/write for software compatibility. The ICH must handle master aborts as if this bit is reset to 0.   |
| 4    | Reserved.   |
| 3    | <b>VGA Enable—R/W.</b><br>1 =Enable. Indicates that the VGA device is on PCI. Therefore, the PCI to hub interface decoder will not accept memory cycles in the range A0000h–BFFFFh. Note that the ICH will never take I/O cycles in the VGA range from PCI.<br>0 =Disable   |
| 2    | <b>ISA Enable—R/W.</b> The ICH ignores this bit. However, this bit is read/write for software compatibility. Since the ICH forwards all I/O cycles that are not in the USB, AC'97, or IDE ranges to PCI, this bit would have no effect.   |
| 1    | <b>SERR# Enable—R/W.</b><br>1 =Enable. If this bit is set AND bit 8 in CMD register (D30:F0 Offset 04h) is also set, the ICH will set the SSE bit in PD_STS register (D30:F0, offset 06h, bit 14) AND also generate an NMI (or SMI# if NMI routed to SMI) when the SERR# signal is asserted.<br>0 =Disable                            |
| 0    | <b>ICH (82801AA):</b><br><b>Parity Error Response Enable—R/W.</b><br>1 = Enable the hub interface to PCI bridge for parity error detection and reporting on the PCI bus. Parity errors on PCI will cause an NMI, if NMI's are enabled.<br>0 = Disable<br><b>ICH0 (82801AB):</b><br>Parity Error Response Enable—RO. Hardwired to '0'. |





### 7.1.26 MTT—Multi-Transaction Timer Register (HUB-PCI—D30:F0)

Offset Address: 70h Attribute: R/W  
 Default Value: 00h Size: 8 bits

MTT is an 8-bit register that controls the amount of time that the ICH’s arbiter allows a PCI initiator to perform multiple back-to-back transactions on the PCI bus. The ICH’s MTT mechanism is used to guarantee a fair share of the Primary PCI bandwidth to an initiator that performs multiple back-to-back transactions to fragmented memory ranges (and as a consequence it can not use long burst transfers).

The number of clocks programmed in the MTT represents the guaranteed time slice (measured in PCI clocks) allotted to the current agent, after which the arbiter will grant another agent that is requesting the bus. The default value of MTT is 00h and disables this function, which could starve some masters from using the PCI bus. Programming of the MTT to anything less than 16 clocks will not allow the Grant-to-FRAME# latency to be 16 clocks. The MTT timer will timeout before the Grant-to-FRAME# trigger causing a re-arbitration.

| Bit | Description  |
|-----|--|
| 7:3 | <b>Multi-Transaction Timer Count Value.</b> This field specifies the amount of time that grant will remain asserted to a master continuously asserting its request for multiple transfers. This field specifies the count in an 8-clock (PCI clock) granularity. |
| 2:0 | Reserved.  |

### 7.1.27 PCI\_MAST\_STS—PCI Master Status Register (HUB-PCI—D30:F0)

Offset Address: 82h Attribute: R/W  
 Default Value: 00h Size: 8 bits

| Bit | Description   |
|-----|---|
| 7:6 | Reserved.   |
| 5:0 | <b>PCI Master Request Status (PCI_MREQ_STS).</b> Allows software to see if a particular bus master has requested use of the PCI bus. This bit is only cleared by writing a 1 to the bit position. |

### 7.1.28 ERR\_CMD—Error Command Register (HUB-PCI—D30:F0)

|                 |     |             |        |
|-----------------|-----|-------------|--------|
| Offset Address: | 90h | Attribute:  | R/W    |
| Default Value:  | 00h | Size:       | 8 bits |
| Lockable:       | No  | Power Well: | Core   |

This register configures the ICH's Device 30 responses to various system errors. The actual assertion of the internal SERR# (routed to cause NMI# or SMI#) is enabled via the PCI Command register.

| Bit | Description   |
|-----|---|
| 7:3 | Reserved.   |
| 2   | <b>SERR# enable on receiving target abort (SERR_RTA_EN).</b><br>1 = Enable. When SERR_EN is set, the ICH will report SERR# when SERR_RTA is set.<br>0 = Disable       |
| 1   | <b>SERR# enable on Delayed Transaction Timeout (SERR_DTT_EN).</b><br>1 = Enable. When SERR_EN is set, the ICH will report SERR# when SERR_DTT is set.<br>0 = Disable. |
| 0   | Reserved.   |

### 7.1.29 ERR\_STS—Error Status Register (HUB-PCI—D30:F0)

|                 |     |             |            |
|-----------------|-----|-------------|------------|
| Offset Address: | 92h | Attribute:  | Read-Write |
| Default Value:  | 00h | Size:       | 8 bits     |
| Lockable:       | No  | Power Well: | Core       |

This register records the cause of system errors in Device 30. The actual assertion of SERR# is enabled via the PCI Command register.

| Bit | Description  |
|-----|--|
| 7:3 | Reserved.  |
| 2   | <b>SERR# Due to Received Target Abort (SERR_RTA).</b><br>1 = The ICH sets this bit when the ICH receives a target abort. If SERR_EN, the ICH will also generate an SERR# when SERR_RTA is set.<br>0 = This bit is cleared by writing a 1.  |
| 1   | <b>SERR# Due to Delayed Transaction Timeout (SERR_DTT).</b><br>1 = When a PCI master does not return for the data within 1 ms of the cycle's completion, the ICH clears the delayed transaction, and sets this bit. If both SERR_DTT_EN and SERR_EN are set, then ICH will also generate an SERR# when SERR_DTT is set.<br>0 = This bit is cleared by writing a 1. |
| 0   | Reserved.  |

# LPC Interface Bridge Registers (D31:F0)

# 8

The LPC Bridge function of the ICH resides in PCI Device 31:Function 0. This function contains many other functional units, such as DMA and Interrupt Controllers, Timers, Power Management, System Management., GPIO, RTC, and LPC Configuration Registers.

In this chapter, registers and functions associated with other functional units (power management, GPIO, USB, IDE, etc.) are described in their respective section

## 8.1 PCI Configuration Registers (D31:F0)

**Note:** Reserved registers are read only and are not shown. Reads to unlisted reserved registers should return all 0's

Table 8-1. PCI Configuration Map (LPC Interface—D31:F0) (Sheet 1 of 2)

| Offset  | Mnemonic    | Register Name                   | Default   | Type |
|---------|-------------|---------------------------------|---|------|
| 00h–01h | VID         | Vendor ID                       | 8086h   | RO   |
| 02h–03h | DID         | Device ID                       | 2410h<br>(ICH: 82801AA)<br>2420h<br>(ICH0: 82801AB) | RO   |
| 04h–05h | PCICMD      | PCI Command Register            | 000Fh   | R/W  |
| 06h–07h | PCISTA      | PCI Device Status Register      | 0280h   | R/W  |
| 08h     | RID         | Revision ID                     | Note 1  | RO   |
| 09h     | PI          | Programming Interface           | 00h   | RO   |
| 0Ah     | SCC         | Sub Class Code                  | 01h   | RO   |
| 0Bh     | BCC         | Base Class Code                 | 06h   | RO   |
| 0Eh     | HEADT       | Header Type                     | 80h   | RO   |
| 40h–43h | PMBASE      | ACPI Base Address Register      | 00000001h   | R/W  |
| 44h     | ACPI_CNTL   | ACPI Control                    | 00h   | R/W  |
| 4Eh–4Fh | BIOS_CNTL   | BIOS Control Register           | 0000h   | R/W  |
| 54h     | TCO_CNTL    | TCO Control                     | 00h   | R/W  |
| 58h–5Bh | GPIO_BASE   | GPIO Base Address Register      | 00000001h   | R/W  |
| 5Ch     | GPIO_CNTL   | GPIO Control Register           | 00h   | R/W  |
| 60h–63h | PIRQ_ROUT   | PIRQ[A-D] Routing Control       | 80h   | R/W  |
| 64h     | SIRQ_CNTL   | Serial IRQ Control Register     | 10h   | R/W  |
| 88h     | D31_ERR_CFG | Device 31 Error Config Register | 00h   | R/W  |
| 8Ah     | D31_ERR_STS | Device 31 Error Status Register | 00h   | R/W  |
| 90h–91h | PCI_DMA_C   | PCI DMA Configuration Registers | 0000h   | R/W  |



### 8.1.3 PCICMD—PCI COMMAND Register (LPC I/F—D31:F0)

|                 |        |             |         |
|-----------------|--------|-------------|---------|
| Offset Address: | 04–05h | Attribute:  | R/W     |
| Default Value:  | 000Fh  | Size:       | 16 bits |
| Lockable:       | No     | Power Well: | Core    |

| Bit   | Description  |
|-------|--|
| 15:10 | Reserved.  |
| 9     | Fast Back to Back Enable (FBE)— RO. Hardwired to '0'.  |
| 8     | <b>SERR_EN</b> (SERR# Enable)— R/W.<br>1 = Enable. Allow SERR# to be generated.<br>0 = Disable.  |
| 7     | Wait Cycle Control (WCC)— RO. Hardwired to '0'.  |
| 6     | <b>ICH (82801AA):</b><br><b>Parity Error Response (PER)</b> — R/W.<br>1 = The 82801AA takes normal action when a parity error is detected. Parity errors on PCI will cause an NMI, if NMI's are enabled.<br>0 = No action is taken when detecting a parity error.<br><b>ICH0 (82801AB):</b><br>Parity Error Response (PER)—RO. Hardwired to '0'. |
| 5     | VGA Palette Snoop (VPS)— RO. Hardwired to '0'  |
| 4     | Postable Memory Write Enable (PMWE)— RO. Hardwired to '0'  |
| 3     | Special Cycle Enable (SCE)—RO. Hardwired to '1'.   |
| 2     | Bus Master Enable (BME)— RO. Hardwired to '1' to indicate that bus mastering can not be disabled for function 0 (DMA/ISA Master)   |
| 1     | Memory Space Enable (MSE)— RO. Hardwired to '1' to indicate that memory space can not be disabled for Function 0 (LPC I/F)   |
| 0     | I/O Space Enable (IOE)— RO. Hardwired to '1' to indicate that the I/O space cannot be disabled for function 0 (LPC I/F)  |

## 8.1.4 PCISTA—PCI Device Status (LPC I/F—D31:F0)

Offset Address: 06–07h Attribute: R/W  
 Default Value: 0280h Size: 16 bits  
 Lockable: No Power Well: Core

| Bit  | Description   |
|------|---|
| 15   | <b>ICH (82801AA):</b><br><b>Detected Parity Error (DPE)— R/W.</b><br>1 = PERR# signal goes active. Set even if the PER bit is 0. This bit is cleared by writing a 1 to this bit position.<br><b>ICH0 (82801AB):</b><br>Detected Parity Error (DPE)—RO. Hardwired to '0'.  |
| 14   | <b>Signaled System Error (SSE)— R/W.</b><br>1 = This bit is set by the ICH if the SERR_EN bit is set and the ICH generates an SERR# on function 0. The ERR_STS register can be read to determine the cause of the SERR#. The SERR# can be routed to cause SMI#, NMI, or interrupt.  |
| 13   | <b>Master Abort Status (RMA)— R/W.</b><br>1 = ICH generated a master abort on PCI due to LPC I/F master or DMA cycles.<br>0 = This bit is cleared by writing a 1 to the bit position.   |
| 12   | <b>Received Target Abort (RTA)— R/W.</b><br>1 = ICH received a target abort during LPC I/F master or DMA cycles to PCI.<br>0 = This bit is cleared by writing a 1 to the bit position.  |
| 11   | <b>Signaled Target Abort (STA)— R/W.</b><br>1 = ICH generated a target abort condition on PCI cycles claimed by the ICH for ICH internal registers or for going to LPC I/F.<br>0 = This bit is cleared by writing a 1 to the bit position.  |
| 10:9 | <b>DEVSEL# Timing Status (DEV_STS)— RO.</b><br>01 = Medium Timing.  |
| 8    | <b>ICH (82801AA):</b><br><b>Data Parity Error Detected (DPED):</b><br>1 = This bit is set when the following conditions occur: <ol style="list-style-type: none"> <li>1. The ICH is the initiator of the cycle,</li> <li>2. The ICH asserted PERR# (for reads) or observed PERR# (for writes), and</li> <li>3. The PER bit is set.</li> </ol> 0 = This bit is reset by writing a 1.<br><b>ICH0 (82801AB):</b><br><b>Data Parity Error Detected (DPED)—RO. Hardwired to '0'.</b> |
| 7    | <b>Fast Back to Back (FB2B)— RO. Always 1. Indicates ICH as a target can accept fast back-to-back transactions.</b>   |
| 6    | <b>User Definable Features (UDF). Hardwired to '0'</b>  |
| 5    | <b>66 MHz Capable (66MHZ_CAP)— RO. Hardwired to '0'</b>   |
| 4:0  | <b>Reserved.</b>  |



### 8.1.10 PMBASE—ACPI Base Address (LPC I/F—D31:F0)

|                 |           |             |              |
|-----------------|-----------|-------------|--------------|
| Offset Address: | 40–43h    | Attribute:  | R/W          |
| Default Value:  | 00000001h | Size:       | 32 bits      |
| Lockable:       | No        | Usage:      | ACPI, Legacy |
|                 |           | Power Well: | Core         |

Sets base address for ACPI I/O registers, GPIO registers and TCO I/O registers. Can be mapped anywhere in the 64K I/O space on 128-byte boundaries.

| Bit   | Description  |
|-------|--|
| 31:16 | Reserved.  |
| 15:7  | <b>Base Address.</b> Provides 128 bytes of I/O space for ACPI, and TCO logic. This is placed on a 128-byte boundary. |
| 6:1   | Reserved.  |
| 0     | <b>Resource Indicator:</b> RO. Tied to 1 to indicate I/O space.  |

### 8.1.11 ACPI\_CNTL—ACPI Control (LPC I/F — D31:F0)

|                 |     |             |              |
|-----------------|-----|-------------|--------------|
| Offset Address: | 44h | Attribute:  | R/W          |
| Default Value:  | 00h | Size:       | 8 bits       |
| Lockable:       | No  | Usage:      | ACPI, Legacy |
|                 |     | Power Well: | Core         |

| Bit  | Description  |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
|------|--|------|---------|-----|------|-----|-------|-----|-------|-----|----------|-----|--|-----|--|-----|--|-----|--|
| 7:5  | Reserved.  |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 4    | <b>ACPI Enable (ACPI_EN).</b><br>1 = Enable. Decode of the I/O range pointed to by the ACPI base register is enabled, and the ACPI power management function is enabled. Note that the APM power management ranges (B2/B3h) are always enabled and are not affected by this bit.<br>0 = Disable.   |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 3    | Reserved.  |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 2:0  | <b>SCI IRQ Select (SCI_IRQ_SEL).</b> Specifies on which IRQ the SCI will internally appear. If not using the APIC, the SCI must be routed to IRQ[9:11], and that interrupt is not sharable with the SERIRQ stream, but is shareable with other PCI interrupts. If using the APIC, the SCI can also be mapped to IRQ[20:23], and can be shared with other interrupts.<br><br><table> <thead> <tr> <th>Bits</th> <th>SCI Map</th> </tr> </thead> <tbody> <tr> <td>000</td> <td>IRQ9</td> </tr> <tr> <td>001</td> <td>IRQ10</td> </tr> <tr> <td>010</td> <td>IRQ11</td> </tr> <tr> <td>011</td> <td>Reserved</td> </tr> <tr> <td>100</td> <td>IRQ20 (Only available if APIC enabled)</td> </tr> <tr> <td>101</td> <td>IRQ21 (Only available if APIC enabled)</td> </tr> <tr> <td>110</td> <td>IRQ22 (Only available if APIC enabled)</td> </tr> <tr> <td>111</td> <td>IRQ23 (Only available if APIC enabled)</td> </tr> </tbody> </table> | Bits | SCI Map | 000 | IRQ9 | 001 | IRQ10 | 010 | IRQ11 | 011 | Reserved | 100 | IRQ20 (Only available if APIC enabled) | 101 | IRQ21 (Only available if APIC enabled) | 110 | IRQ22 (Only available if APIC enabled) | 111 | IRQ23 (Only available if APIC enabled) |
| Bits | SCI Map  |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 000  | IRQ9   |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 001  | IRQ10  |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 010  | IRQ11  |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 011  | Reserved   |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 100  | IRQ20 (Only available if APIC enabled)   |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 101  | IRQ21 (Only available if APIC enabled)   |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 110  | IRQ22 (Only available if APIC enabled)   |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 111  | IRQ23 (Only available if APIC enabled)   |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |



### 8.1.12 BIOS\_CNTL (LPC I/F—D31:F0)

|                 |        |             |         |
|-----------------|--------|-------------|---------|
| Offset Address: | 4E–4Fh | Attribute:  | R/W     |
| Default Value:  | 0000h  | Size:       | 16 bits |
| Lockable:       | No     | Power Well: | Core    |

| Bit  | Description   |
|------|---|
| 15:2 | Reserved.   |
| 1    | <b>BIOS Lock Enable (BLE).</b> Once set, this bit can only be cleared by a PCIRST#.<br>1 = Setting the BIOSWE bit will cause SMIs.<br>0 = Setting the BIOSWE will not cause SMIs.   |
| 0    | <b>BIOS Write Enable (BIOSWE).</b> When this bit is written from a '0' to a '1' and BIOS lock Enable (BLE) is also set, an SMI# is generated. This ensures that only SMM code can update BIOS.<br>1 = Access to the BIOS space is enabled for both read and write cycles.<br>0 = Only read cycles result in LPC I/F cycles. |

### 8.1.13 TCO\_CNTL — TCO Control (LPC I/F — D31:F0)

|                 |     |             |        |
|-----------------|-----|-------------|--------|
| Offset Address: | 54h | Attribute:  | R/W    |
| Default Value:  | 00h | Size:       | 8 bits |
| Lockable:       | No  | Power Well: | Core   |

| Bit  | Description   |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
|------|---|------|---------|-----|------|-----|-------|-----|-------|-----|----------|-----|--|-----|--|-----|--|-----|--|
| 7:4  | Reserved.   |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 3    | <b>TCO Interrupt Enable (TCO_INT_EN).</b> This bit enables/disables the TCO interrupt.<br>1 = Enables TCO Interrupt, as selected by the TCO_INT_SEL field.<br>0 = Disables TCO interrupt.   |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 2:0  | <b>TCO Interrupt Select (TCO_INT_SEL).</b> Specifies on which IRQ the TCO will internally appear. If not using the APIC, the TCO interrupt must be routed to IRQ[9:11], and that interrupt is not sharable with the SERIRQ stream, but is shareable with other PCI interrupts. If using the APIC, the TCO interrupt can also be mapped to IRQ[20:23], and can be shared with other interrupt. Note that if the TCOSCI_EN bit is set (bit 6 in this register), then the TCO interrupt will be sent to the same interrupt as the SCI, and the TCO_INT_SEL bits will have no meaning.<br><table border="0"> <thead> <tr> <th>Bits</th> <th>SCI Map</th> </tr> </thead> <tbody> <tr> <td>000</td> <td>IRQ9</td> </tr> <tr> <td>001</td> <td>IRQ10</td> </tr> <tr> <td>010</td> <td>IRQ11</td> </tr> <tr> <td>011</td> <td>Reserved</td> </tr> <tr> <td>100</td> <td>IRQ20 (Only available if APIC enabled)</td> </tr> <tr> <td>101</td> <td>IRQ21 (Only available if APIC enabled)</td> </tr> <tr> <td>110</td> <td>IRQ22 (Only available if APIC enabled)</td> </tr> <tr> <td>111</td> <td>IRQ23 (Only available if APIC enabled)</td> </tr> </tbody> </table> | Bits | SCI Map | 000 | IRQ9 | 001 | IRQ10 | 010 | IRQ11 | 011 | Reserved | 100 | IRQ20 (Only available if APIC enabled) | 101 | IRQ21 (Only available if APIC enabled) | 110 | IRQ22 (Only available if APIC enabled) | 111 | IRQ23 (Only available if APIC enabled) |
| Bits | SCI Map   |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 000  | IRQ9  |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 001  | IRQ10   |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 010  | IRQ11   |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 011  | Reserved  |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 100  | IRQ20 (Only available if APIC enabled)  |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 101  | IRQ21 (Only available if APIC enabled)  |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 110  | IRQ22 (Only available if APIC enabled)  |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |
| 111  | IRQ23 (Only available if APIC enabled)  |      |         |     |      |     |       |     |       |     |          |     |  |     |  |     |  |     |  |

**8.1.14 GPIOBASE—GPIO Base Address (LPC I/F—D31:F0)**

|                 |           |             |         |
|-----------------|-----------|-------------|---------|
| Offset Address: | 58h–5Bh   | Attribute:  | R/W     |
| Default Value:  | 00000001h | Size:       | 32 bits |
| Lockable:       | No        | Power Well: | Core    |

| Bit   | Description   |
|-------|---|
| 31:16 | Reserved.   |
| 15:6  | <b>Base Address.</b> Provides the 64 bytes of I/O space for GPIO. |
| 5:1   | Reserved.   |
| 0     | <b>Resource Indicator.</b> Tied to 1 to indicate I/O space.       |

**8.1.15 GPIO\_CNTL—GPIO Control (LPC I/F—D31:F0)**

|                 |     |             |        |
|-----------------|-----|-------------|--------|
| Offset Address: | 5Ch | Attribute:  | R/W    |
| Default Value:  | 00h | Size:       | 8 bits |
| Lockable:       | No  | Power Well: | Core   |

| Bit | Description   |
|-----|---|
| 7:5 | Reserved.   |
| 4   | <b>GPIO Enable (GPIO_EN).</b> This bit enables/disables decode of the I/O range pointed to by the GPIO base register and enables/disables the GPIO function.<br>1 = Enable<br>0 = Disable |
| 3:0 | Reserved.   |

### 8.1.16 PIRQ[n]\_ROUT—PIRQ[A,B,C,D] Routing Control (LPC I/F—D31:F0)

Offset Address: PIRQA–60h, PIRQB–61h, PIRQC–62h, PIRQD–63h      Attribute: R/W  
 Default Value: 80h      Size: 8 bits  
 Lockable: No      Power Well: Core

| Bit | Description   |
|-----|---|
| 7   | <b>Interrupt Routing Enable (IRQEN).</b><br>0 = The corresponding PIRQ is routed to one of the ISA-compatible interrupts specified in bits[3:0].<br>1 = The PIRQ is not routed to the 8259.<br>Note: If the PIRQ is intended to cause an interrupt to the ICH's integrated I/O APIC, then this bit should be set to 1 and the APIC_EN bit should be set to 0 (mask interrupt to the 8259 and unmask interrupt to the I/O APIC). In that case, the PIRQ[A:D] signals are routed to IRQ16:19 in the I/O APIC. |
| 6:4 | Reserved.   |
| 3:0 | <b>IRQ Routing. (ISA compatible).</b><br>0000 = Reserved      1000 = Reserved<br>0001 = Reserved      1001 = IRQ9<br>0010 = Reserved      1010 = IRQ10<br>0011 = IRQ3      1011 = IRQ11<br>0100 = IRQ4      1100 = IRQ12<br>0101 = IRQ5      1101 = Reserved<br>0110 = IRQ6      1110 = IRQ14<br>0111 = IRQ7      1111 = IRQ15  |

### 8.1.17 SERIRQ\_CNTL—Serial IRQ Control (LPC I/F—D31:F0)

Offset Address: 64h      Attribute: R/W  
 Default Value: 10h      Size: 8 bits  
 Lockable: No      Power Well: Core

| Bit | Description   |
|-----|---|
| 7   | <b>Serial IRQ Enable (SIRQEN).</b><br>1 = Serial IRQs will be recognized. The SERIRQ pin will be configured as SERIRQ.<br>0 = The buffer is input only and internally SERIRQ will be a '1'.   |
| 6   | <b>Serial IRQ Mode Select (SIRQMD).</b> For systems using Quiet Mode, this bit should be set to 1 (Continuous Mode) for at least one frame after coming out of reset before switching back to Quiet Mode. Failure to do so will result in the ICH not recognizing SERIRQ interrupts.<br>1 = The serial IRQ machine will be in continuous mode.<br>0 = The serial IRQ machine will be in quiet mode.   |
| 5:2 | <b>Serial IRQ Frame Size (SIRQSZ).</b> Fixed field that indicates the size of the SERIRQ frame. In the ICH, this field needs to be programmed to 21 frames (0100). This is an offset from a base of 17 which is the smallest data frame size.   |
| 1:0 | <b>Start Frame Pulse Width (SFPW).</b> This is the number of PCI clocks that the SERIRQ pin will be driven low by the serial IRQ machine to signal a start frame. In continuous mode, the ICH will drive the start frame for the number of clocks specified. In quiet mode, the ICH will drive the start frame for the number of clocks specified minus one, as the first clock was driven by the peripheral.<br>00 = 4 clocks<br>01 = 6 clocks<br>10 = 8 clocks<br>11 = Reserved |

### 8.1.18 D31\_ERR\_CFG—Device 31 Error Config Register (LPC I/F—D31:F0)

|                 |     |             |        |
|-----------------|-----|-------------|--------|
| Offset Address: | 88h | Attribute:  | R/W    |
| Default Value:  | 00h | Size:       | 8 bits |
| Lockable:       | No  | Power Well: | Core   |

This register configures the ICH's Device 31 responses to various system errors. The actual assertion of SERR# is enabled via the PCI Command register

| Bit | Description  |
|-----|--|
| 7:3 | Reserved.  |
| 2   | <b>SERR_RTA_EN.</b> Enable for SERR# on receiving target abort.<br>1 = When this bit is 1 and SERR_EN is set, the ICH will generate SERR# when SERR_RTA is set.                |
| 1   | <b>SERR_DTT_EN.</b> Enable for SERR# on Delayed Transaction Timeout.<br>1 = When this bit is set and the SERR_EN is set, the ICH will generate SERR# when SERR_DTT bit is set. |
| 0   | Reserved   |

### 8.1.19 D31\_ERR\_STS—Device 31 Error Status Register (LPC I/F—D31:F0)

|                 |     |             |            |
|-----------------|-----|-------------|------------|
| Offset Address: | 8Ah | Attribute:  | Read-Write |
| Default Value:  | 00h | Size:       | 8 bits     |
| Lockable:       | No  | Power Well: | Core       |

This register configures the ICH's Device 31 responses to various system errors. The actual assertion of SERR# is enabled via the PCI Command register.

| Bit | Description   |
|-----|---|
| 7:3 | Reserved.   |
| 2   | <b>SERR_RTA.</b> SERR# Due to Received Target Abort .<br>1 = The ICH sets this bit when it receives a target abort. If SERR_EN, the ICH will also generate an SERR# when SERR_RTA is set. This bit is cleared by writing a 1 back to this bit.  |
| 1   | <b>SERR_DTT.</b> SERR# Due to Delayed Transaction Timeout .<br>1 = When a PCI master does not return for the data within 1 ms of the cycle's completion, the ICH clears the delayed transaction and sets this bit. If both SERR_DTT_EN and SERR_EN are set, then ICH will also generate an SERR# when SERR_DTT is set. This bit is cleared by writing a 1 back to this bit. |
| 0   | Reserved.   |

### 8.1.20 PCI\_DMA\_CFG—PCI DMA Configuration (LPC I/F—D31:F0)

|                 |         |             |         |
|-----------------|---------|-------------|---------|
| Offset Address: | 90h–91h | Attribute:  | R/W     |
| Default Value:  | 0000h   | Size:       | 16 bits |
| Lockable:       | No      | Power Well: | Core    |

| Bit   | Description   |
|-------|---|
| 15:14 | <b>Channel 7 Select.</b><br>00 = Reserved<br>01 = PC/PCI DMA<br>10 = Reserved<br>11 = LPC I/F DMA |
| 13:12 | <b>Channel 6 Select.</b> Same bit decode as for Channel 7   |
| 11:10 | <b>Channel 5 Select.</b> Same bit decode as for Channel 7   |
| 9:8   | Reserved.   |
| 7:6   | <b>Channel 3 Select.</b> Same bit decode as for Channel 7   |
| 5:4   | <b>Channel 2 Select.</b> Same bit decode as for Channel 7   |
| 3:2   | <b>Channel 1 Select.</b> Same bit decode as for Channel 7   |
| 1:0   | <b>Channel 0 Select.</b> Same bit decode as for Channel 7   |

### 8.1.21 GEN\_CNTL — General Control Register (LPC I/F — D31:F0)

|                 |   |             |         |
|-----------------|---|-------------|---------|
| Offset Address: | D0h–D3h   | Attribute:  | R/W     |
| Lockable:       | No  | Size:       | 32 bits |
| Default Value:  | 00000000h (ICH; 82801AA)<br>02000000h (ICH0; 82901AB) | Power Well: | Core    |

| Bit   | Description   |
|-------|---|
| 31:26 | Reserved.   |
| 25    | <b>ICH (82801AA):</b><br><b>PCPCIB_SEL.</b><br>1 = The PCI REQ[5]/GNT[5]# pair will, instead, be used for PC/PCI REQ[B]/GNT[B]#. The corresponding bits in the GPIO USE_SEL register must also be set to 0. If the corresponding bits in the GPIO USE_SEL register are set to 1, then the signals will be used as a GPI and GPO<br><b>ICH0 (82801AB):</b><br>Reserved   |
| 24    | <b>HIDE_MISA.</b><br>1 = Software sets this bit to 1 to disable config cycle from being claimed by the 82380AB PCI-TO-ISA Bridge (MISA). This will prevent the OS PCI PnP from getting confused by seeing two ISA bridges.<br>It is required for the ICH PCI address line AD22 to connect to the MISA IDSEL. When this bit is set, the ICH will not assert AD22 during config cycles to the MISA.<br>0 = The ICH will not prevent AD22 from asserting during config cycles to the MISA. |
| 23:14 | Reserved.   |

| Bit  | Description   |
|------|---|
| 13   | <b>Coprocessor Error Enable (COPR_ERR_EN).</b><br>1 = When FERR# is low, ICH generates IRQ13 internally and holds it until an I/O write to port F0h. It will also drive IGNNE# active.<br>0 = FERR# will not generate IRQ13 nor IGNNE#.   |
| 12   | <b>Keyboard IRQ1 Latch Enable (IRQ1LEN).</b><br>1 = The active edge of IRQ1 will be latched and held until a port 60h read.<br>0 = IRQ1 will bypass the latch.  |
| 11   | <b>Mouse IRQ12 Latch Enable (IRQ12LEN).</b><br>1 = The active edge of IRQ12 will be latched and held until a port 60h read.<br>0 = IRQ12 will bypass the latch.   |
| 10:9 | Reserved  |
| 8    | <b>APIC Enable (APIC_EN).</b><br>1 = Enables the internal I/O APIC and its address decode.<br>0 = Disables internal I/O APIC.   |
| 7    | Reserved.   |
| 6    | <b>Alternate Access Mode Enable (ALTACC_EN).</b><br>1 = Alt Access Mode Enable<br>0 = Alt Access Mode Disabled (default). Alt Access Mode allows reads to otherwise unreadable registers and writes otherwise unwritable registers.   |
| 5:3  | Reserved.   |
| 2    | <b>DMA Collection Buffer Enable (DCB_EN).</b><br>1 = Enables DMA Collection Buffer (DCB) for LPC I/F and PC/PCI DMA.<br>0 = DCB disabled.   |
| 1    | <b>Delayed Transaction Enable (DTE).</b><br>1 = ICH enables delayed transactions for internal register, FWH and LPC I/F accesses.<br>0 = delayed transactions disabled.   |
| 0    | <b>Positive Decode Enable (POS_DEC_EN).</b><br>1 = Enables ICH to only perform positive decode on the PCI bus.<br>0 = The ICH will perform subtractive decode on the PCI bus and forward the cycles to LPC I/F if not to an internal register or other known target on LPC I/F. Accesses to internal registers and to known LPC I/F devices will still be positively decoded. |

### 8.1.22 GEN\_STA—General Status (LPC I/F—D31:F0)

|                 |          |             |                       |
|-----------------|----------|-------------|-----------------------|
| Offset Address: | D4h–D7h  | Attribute:  | R/W                   |
| Default Value:  | 0000F0Xh | Size:       | 32 bits               |
| Lockable:       | No       | Power Well: | Core(0:7), RTC (8:15) |

| Bit   | Description  |
|-------|--|
| 31:13 | Reserved.  |
| 12    | <p><b>ICH (82801AA):</b><br/> <b>Enables Processor BIST (CPU_BIST_EN)—R/W.</b><br/>                     1 = Enable. If this bit is set, then the INIT# signal will be driven active when CPURST# is active. INIT# will go inactive with the same timings as the other processor interface signals (Hold Time after CPURST# inactive). Note that CPURST# is generated by the host controller, but the ICH has a hub interface special cycle that allows the ICH to control the assertion/deassertion of CPURST#.</p> <p>0 = Disable</p> <p><b>ICH0 (82801AB):</b><br/>                     Reserved</p> |
| 11:8  | <p><b>FREQ_STRAP[3:0]—R/W:</b> These bits determine the internal frequency multiplier of the processor. These bits can be reset to 1111 based on an external pin strap or via the RTCRST# input signal. Software must program this field based on the processor's specified frequency. These bits are in the RTC well.</p>   |
| 7:3   | Reserved   |
| 2     | <p><b>SAFE_MODE—RO.</b><br/>                     1 = ICH sampled AC_SDOUT high on the rising edge of PWROK. ICH will force FREQ_STRAP[3:0] bits to all 1's (safe mode multiplier).<br/>                     0 = ICH sampled AC_SDOUT low on the rising edge of PWROK.</p>  |
| 1     | <p><b>NO_REBOOT—R/W.</b><br/>                     1 = ICH will disable the TCO Timer system reboot feature. This bit is set either by hardware when SPKR is sampled low on the rising edge of PWROK, or by software writing a 1 to the bit.</p>  |
| 0     | Reserved   |





### 8.1.25 FDD/LPT\_DEC—LPC I/F FDD & LPT Decode Ranges (LPC I/F—D31:F0)

|                |     |             |            |
|----------------|-----|-------------|------------|
| Offset:        | E1h | Attribute:  | Read/Write |
| Default Value: | 00h | Size:       | 8 bits     |
| Lockable:      | No  | Power Well: | Core       |

| Bit | Description   |
|-----|---|
| 7:5 | Reserved  |
| 4   | <b>FDD Decode Range.</b> The following list describes which range to decode for the FDD Port<br>0 = 3F0h–3F5h, 3F7h (Primary)<br>1 = 370h–2FFh (Secondary)  |
| 3:2 | Reserved  |
| 1:0 | <b>LPT Decode Range.</b> The following list describes which range to decode for the LPTPort<br>00 = 378h–37Fh and 778h–77Fh<br>01 = 278h–27Fh (port 279h is read only) and 678h–67Fh<br>10 = 3BCh–3BEh and 7BCh–7BEh<br>11 = Reserved |

### 8.1.26 SND\_DEC—LPC I/F Sound Decode Ranges (LPC I/F—D31:F0)

|                |     |             |            |
|----------------|-----|-------------|------------|
| Offset:        | E2h | Attribute:  | Read/Write |
| Default Value: | 00h | Size:       | 8 bits     |
| Lockable:      | No  | Power Well: | Core       |

| Bit | Description  |
|-----|--|
| 7:6 | Reserved   |
| 5:4 | <b>MSS Decode Range.</b> The following list describes which range to decode for the Microsoft Sound System (MSS).<br>00 = 530h–537h<br>01 = 604h–60Bh<br>10 = E80h–E87h<br>11 = F40h–F47h  |
| 3   | <b>MIDI Decode Range.</b> The following list describes which range to decode for the Midi Port.<br>0 = 330h–331h<br>1 = 300h–301h  |
| 2   | Reserved   |
| 1:0 | <b>SB16 Decode Range.</b> The following list describes which range to decode for the Sound Blaster 16 (SB16) Port.<br>00 = 220h–233h<br>01 = 240h–253h<br>10 = 260h–273h<br>11 = 280h–293h |

### 8.1.27 GEN1\_DEC—LPC I/F Generic Decode Range 1 (LPC I/F—D31:F0)

Offset: E4h–E5h Attribute: Read/Write  
 Default Value: 00h Size: 16 bits  
 Lockable: Yes Power Well: Core

| Bit  | Description  |
|------|--|
| 15:7 | <b>GEN1_BASE.</b> Base address for this generic I/O decode range. This address is aligned on a 128-byte boundary and must have address lines 31:16 as 0.<br>Note that this generic decode is for I/O addresses only, not memory addresses. |
| 6:1  | Reserved. Read as 0  |
| 0    | <b>GEN1_EN.</b><br>1 = Enable. Enables the GEN1 I/O range to be forwarded to the LPC I/F.<br>0 = Disable   |

### 8.1.28 GEN2\_DEC—LPC I/F Generic Decode Range 2 (LPC I/F—D31:F0)

Offset: ECh–EDh Attribute: Read/Write  
 Default Value: 00h Size: 16 bits  
 Lockable: Yes Power Well: Core

| Bit  | Description  |
|------|--|
| 15:4 | <b>GEN2_BASE.</b> Base address for this generic I/O decode range. This address is aligned on a 16-byte boundary, and must have address lines 31:16 as 0.<br>Note that this generic decode is for I/O addresses only, not memory addresses. |
| 3:1  | Reserved. Read as 0  |
| 0    | <b>GEN2_EN.</b><br>1 = Enable. Enables the GEN2 I/O range to be forwarded to the LPC I/F.<br>0 = Disable.  |

### 8.1.29 LPC\_EN—LPC I/F Enables (LPC I/F—D31:F0)

|                |         |             |            |
|----------------|---------|-------------|------------|
| Offset:        | E6h–E7h | Attribute:  | Read/Write |
| Default Value: | 00h     | Size:       | 16 bits    |
| Lockable:      | Yes     | Power Well: | Core       |

For the bits in this register 1 = Enable and 0 = Disable.

| Bit   | Description  |
|-------|--|
| 15:14 | Reserved   |
| 13    | <b>CNF2_LPC_EN:</b> This bit enables the decoding of the I/O locations 4Eh and 4Fh to the LPC interface. This range is used for a microcontroller.       |
| 12    | <b>CNF1_LPC_EN:</b> This bit enables the decoding of the I/O locations 2Eh and 2Fh to the LPC interface. This range is used for Super I/O devices.       |
| 11    | <b>MC_LPC_EN:</b> This bit enables the decoding of the I/O locations 62h and 66h to the LPC interface. This range is used for a MicroController.         |
| 10    | <b>KBC_LPC_EN:</b> This bit enables the decoding of the I/O locations 60h and 64h to the LPC interface. This range is used for a microcontroller.        |
| 9     | <b>GAMEH_LPC_EN:</b> This bit enables the decoding of the I/O locations 208h to 20Fh to the LPC interface. This range is used for a gameport.            |
| 8     | <b>GAMEL_LPC_EN:</b> This bit enables the decoding of the I/O locations 200h to 207h to the LPC interface. This range is used for a gameport.            |
| 7     | <b>ADLIB_LPC_EN:</b> This bit enables the decoding of the I/O locations 388h–38Bh to the LPC interface.  |
| 6     | <b>MSS_LPC_EN:</b> This bit enables the decoding of the MSS range to the LPC interface. This range is selected in the LPC_Sound Decode Range Register.   |
| 5     | <b>MIDI_LPC_EN:</b> This bit enables the decoding of the MIDI range to the LPC interface. This range is selected in the LPC_Sound Decode Range Register. |
| 4     | <b>SB16_LPC_EN:</b> This bit enables the decoding of the SB16 range to the LPC interface. This range is selected in the LPC_Sound Decode Range Register. |
| 3     | <b>FDD_LPC_EN:</b> This bit enables the decoding of the FDD range to the LPC interface. This range is selected in the LPC_FDD/LPT Decode Range Register. |
| 2     | <b>LPT_LPC_EN:</b> This bit enables the decoding of the LPT range to the LPC interface. This range is selected in the LPC_FDD/LPT Decode Range Register. |
| 1     | <b>COMB_LPC_EN:</b> This bit enables the decoding of the COMB range to the LPC interface. This range is selected in the LPC_COM Decode Range Register.   |
| 0     | <b>COMA_LPC_EN:</b> This bit enables the decoding of the COMA range to the LPC interface. This range is selected in the LPC_COM Decode Range Register.   |

### 8.1.30 FWH\_DEC\_EN—FWH Decode Enable Register (LPC I/F—D31:F0)

Offset Address: E3h Attribute: R/W  
 Default Value: 80h Size: 8 bits

| Bit | Description   |
|-----|---|
| 7   | <p><b>FWH_F8_EN:</b> (Hardwired to 1) Enables decoding two 512 KB FWH memory ranges, and one 128 KB memory range.</p> <p>1 = Enable the following ranges for the FWH<br/>           FFF80000h–FFFFFFFFh<br/>           FFB80000h–FFBFFFFFFh<br/>           000E0000h–000FFFFFFh</p> |
| 6   | <p><b>FWH_F0_EN:</b> Enables decoding two 512 KB FWH memory ranges.</p> <p>1 = Enable the following ranges for the FWH<br/>           FFF00000h–FFF7FFFFh<br/>           FFB00000h–FFB7FFFFh</p> <p>0 = Disable</p>   |
| 5   | <p><b>FWH_E8_EN:</b> Enables decoding two 512 KB FWH memory ranges.</p> <p>1 = Enable the following ranges for the FWH<br/>           FFE80000h–FFEFFFFFFh<br/>           FFA80000h–FFAFFFFFFh</p> <p>0 = Disable</p>   |
| 4   | <p><b>FWH_E0_EN:</b> Enables decoding two 512 KB FWH memory ranges.</p> <p>1 = Enable the following ranges for the FWH<br/>           FFE00000h–FFE7FFFFh<br/>           FFA00000h–FFA7FFFFh</p> <p>0 = Disable</p>   |
| 3   | <p><b>FWH_D8_EN:</b> Enables decoding two 512 KB FWH memory ranges.</p> <p>1 = Enable the following ranges for the FWH<br/>           FFD80000h–FFDFFFFFFh<br/>           FF980000h–FF9FFFFFFh</p> <p>0 = Disable</p>   |
| 2   | <p><b>FWH_D0_EN:</b> Enables decoding two 512 KB FWH memory ranges.</p> <p>1 = Enable the following ranges for the FWH<br/>           FFD00000h–FFD7FFFFh<br/>           FF900000h–FF97FFFFh</p> <p>0 = Disable</p>   |
| 1   | <p><b>FWH_C8_EN:</b> Enables decoding two 512 KB FWH memory ranges.</p> <p>1 = Enable the following ranges for the FWH<br/>           FFC80000h–FFCFFFFFFh<br/>           FF880000h–FF8FFFFFFh</p> <p>0 = Disable</p>   |
| 0   | <p><b>FWH_C0_EN:</b> Enables decoding two 512 KB FWH memory ranges.</p> <p>1 = Enable the following ranges for the FWH<br/>           FFC00000h–FFC7FFFFh<br/>           FF800000h–FF87FFFFh</p> <p>0 = Disable</p>   |



### 8.1.32 FUNC\_DIS—Function Disable Register (LPC I/F—D31:F0)

Offset: F2h–F3h Attribute: Read/Write  
 Default Value: 0000h Size: 16 bits  
 Lockable: No Power Well: Core

| Bit  | Description   |
|------|---|
| 15:9 | <b>Reserved</b>   |
| 8    | <b>SMB_FOR_BIOS:</b> This bit is used in conjunction with bit 3 in this register.<br>1 = Allows the SMBus I/O space to be accessible by software when bit 3 in this register is set. The PCI configuration space is hidden in this case.<br>Note that if bit 3 is set alone, the decode of both SMBus PCI configuration and I/O space will be disabled.<br>0 = No effect.   |
| 7    | Reserved  |
| 6    | <b>F6_Disable:</b> Software sets this bit to disable the AC'97 modem controller function. When disabled, the ICH does not decode the PCI config space registers, I/O ranges, or memory ranges associated with the AC'97 modem controller. When this bit is set, the AC'97 modem controller will not cause any interrupts or power management events.<br>0 = AC'97 Modem is enabled<br>1 = AC'97 Modem is disabled                       |
| 5    | <b>F5_Disable:</b> Software sets this bit to disable the AC'97 audio controller function. When disabled, the ICH does not decode the PCI config space registers, I/O ranges, or memory ranges associated with the AC'97 audio controller. When this bit is set, the AC'97 audio controller will not cause any interrupts or power management events.<br>0 = AC'97 audio controller is enabled<br>1 = AC'97 audio controller is disabled |
| 4    | Reserved.   |
| 3    | <b>F3_Disable:</b> Software sets this bit to disable the SMBus Host Controller function. When disabled, the ICH does not decode the PCI config space registers, I/O ranges, or memory ranges associated with the SMBus controller. When this bit is set, the SMBus controller will not cause any interrupts or power management events.<br>0 = SMBus controller is enabled<br>1 = SMBus controller is disabled                          |
| 2    | <b>F2_Disable:</b> Software sets this bit to disable the USB controller function. When disabled, the ICH does not decode the PCI config space registers, I/O ranges, or memory ranges associated with the USB controller. When this bit is set, the USB controller will not cause any interrupts or power management events.<br>0 = USB controller is enabled<br>1 = USB controller is disabled   |
| 1    | <b>F1_Disable:</b> Software sets this bit to disable the IDE controller function. When disabled, the ICH does not decode the PCI config space registers, I/O ranges, or memory ranges associated with the IDE controller. When this bit is set, the IDE controller will not cause any interrupts or power management events.<br>0 = IDE controller is enabled<br>1 = IDE controller is disabled   |
| 0    | Reserved.   |

## 8.2 DMA I/O Registers

Table 8-2. DMA Registers (Sheet 1 of 2)

| Port    | Alias   | Register Name/Function                        | Default   | Type |
|---------|---------|---|-----------|------|
| 00h     | 10h     | Channel 0 DMA Base & Current Address Register | Undefined | R/W  |
| 01h     | 11h     | Channel 0 DMA Base & Current Count Register   | Undefined | R/W  |
| 02h     | 12h     | Channel 1 DMA Base & Current Address Register | Undefined | R/W  |
| 03h     | 13h     | Channel 1 DMA Base & Current Count Register   | Undefined | R/W  |
| 04h     | 14h     | Channel 2 DMA Base & Current Address Register | Undefined | R/W  |
| 05h     | 15h     | Channel 2 DMA Base & Current Count Register   | Undefined | R/W  |
| 06h     | 16h     | Channel 3 DMA Base & Current Address Register | Undefined | R/W  |
| 07h     | 17h     | Channel 3 DMA Base & Current Count Register   | Undefined | R/W  |
| 08h     | 18h     | Channel 0–3 DMA Command Register              | Undefined | WO   |
|         |         | Channel 0–3 DMA Status Register               | Undefined | RO   |
| 0Ah     | 1Ah     | Channel 0–3 DMA Write Single Mask Register    | 000001XXb | WO   |
| 0Bh     | 1Bh     | Channel 0–3 DMA Channel Mode Register         | 000000XXb | WO   |
| 0Ch     | 1Ch     | Channel 0–3 DMA Clear Byte Pointer Register   | Undefined | WO   |
| 0Dh     | 1Dh     | Channel 0–3 DMA Master Clear Register         | Undefined | WO   |
| 0Eh     | 1Eh     | Channel 0–3 DMA Clear Mask Register           | Undefined | WO   |
| 0Fh     | 1Fh     | Channel 0–3 DMA Write All Mask Register       | 0Fh       | R/W  |
| 80h     | 90h     | Reserved Page Register                        | Undefined | R/W  |
| 81h     | 91h     | Channel 2 DMA Memory Low Page Register        | Undefined | R/W  |
| 82h     | -       | Channel 3 DMA Memory Low Page Register        | Undefined | R/W  |
| 83h     | 93h     | Channel 1 DMA Memory Low Page Register        | Undefined | R/W  |
| 84h–86h | 94h–96h | Reserved Page Registers                       | Undefined | R/W  |
| 87h     | 98h     | Channel 0 DMA Memory Low Page Register        | Undefined | R/W  |
| 88h     | 98h     | Reserved Page Register                        | Undefined | R/W  |
| 89h     | 99h     | Channel 6 DMA Memory Low Page Register        | Undefined | R/W  |
| 8Ah     | 9Ah     | Channel 7 DMA Memory Low Page Register        | Undefined | R/W  |
| 8Bh     | 9Bh     | Channel 5 DMA Memory Low Page Register        | Undefined | R/W  |
| 8Ch–8Eh | 9Ch–9Eh | Reserved Page Registers                       | Undefined | R/W  |
| 8Fh     | 9Fh     | Refresh Low Page Register                     | Undefined | R/W  |
| C0h     | C1h     | Channel 4 DMA Base & Current Address Register | Undefined | R/W  |
| C2h     | C3h     | Channel 4 DMA Base & Current Count Register   | Undefined | R/W  |
| C4h     | C5h     | Channel 5 DMA Base & Current Address Register | Undefined | R/W  |
| C6h     | C7h     | Channel 5 DMA Base & Current Count Register   | Undefined | R/W  |
| C8h     | C9h     | Channel 6 DMA Base & Current Address Register | Undefined | R/W  |
| CAh     | CBh     | Channel 6 DMA Base & Current Count Register   | Undefined | R/W  |
| CCh     | CDh     | Channel 7 DMA Base & Current Address Register | Undefined | R/W  |

Table 8-2. DMA Registers (Sheet 2 of 2)

|     |     |   |           |     |
|-----|-----|---|-----------|-----|
| CEh | CFh | Channel 7 DMA Base & Current Count Register | Undefined | R/W |
| D0h | D1h | Channel 4–7 DMA Command Register            | Undefined | WO  |
|     |     | Channel 4–7 DMA Status Register             | Undefined | RO  |
| D4h | D5h | Channel 4–7 DMA Write Single Mask Register  | 000001XXb | WO  |
| D6h | D7h | Channel 4–7 DMA Channel Mode Register       | 000000XXb | WO  |
| D8h | D9h | Channel 4–7 DMA Clear Byte Pointer Register | Undefined | WO  |
| DAh | DBh | Channel 4–7 DMA Master Clear Register       | Undefined | WO  |
| DCh | DDh | Channel 4–7 DMA Clear Mask Register         | Undefined | WO  |
| DEh | DFh | Channel 4–7 DMA Write All Mask Register     | 0Fh       | R/W |

### 8.2.1 DMABASE\_CA—DMA Base and Current Address Registers

I/O Address: Ch. #0 = 00h; Ch. #1 = 02h Attribute: RO  
 Ch. #2 = 04h; Ch. #3 = 06h Size: 16 bits (per channel),  
 Ch. #5 = C4h; Ch. #6 = C8h but accessed in two 8 bits  
 Ch. #7 = CCh; quantities  
 Default Value: Undef  
 Lockable: No Power Well: Core

| Bit  | Description   |
|------|---|
| 15:0 | <p><b>Base and Current Address.</b> This register determines the address for the transfers to be performed. The address specified points to two separate registers. On writes, the value is stored in the <i>Base Address</i> register and copied to the <i>Current Address</i> register. On reads, the value is returned from the <i>Current Address</i> register.</p> <p>The address increments/decrements in the Current Address register after each transfer, depending on the mode of the transfer. If the channel is in auto-initialize mode, the Current Address register will be reloaded from the Base Address register after a terminal count is generated.</p> <p>For transfers to/from a 16-bit slave (channel's 5-7), the address is shifted left one bit location. Bit 15 will be shifted out. Therefore, if bit 15 was a '1', it will be lost.</p> <p>The register is accessed in 8 bit quantities. The byte is pointed to by the current byte pointer flip/flop. Before accessing an address register, the byte pointer flip/flop should be cleared to ensure that the low byte is accessed first</p> |



## 8.2.2 DMABASE\_CC—DMA Base and Current Count Registers

|                |   |             |  |
|----------------|---|-------------|--|
| I/O Address:   | Ch. #0 = 01h; Ch. #1 = 03h<br>Ch. #2 = 05h; Ch. #3 = 07h<br>Ch. #5 = C6h; Ch. #6 = CAh<br>Ch. #7 = CEh; | Attribute:  | R/W  |
|                |   | Size:       | 16 bits (per channel),<br>but accessed in two 8 bits<br>quantities |
| Default Value: | Undefined   |             |  |
| Lockable:      | No  | Power Well: | Core   |

| Bit  | Description  |
|------|--|
| 15:0 | <p><b>Base and Current Count.</b> This register determines the number of transfers to be performed. The address specified points to two separate registers. On writes, the value is stored in the <i>Base Count</i> register and copied to the <i>Current Count</i> register. On reads, the value is returned from the <i>Current Count</i> register.</p> <p>The actual number of transfers is one more than the number programmed in the Base Count Register (i.e., programming a count of 4h results in 5 transfers). The count is decrements in the Current Count register after each transfer. When the value in the register rolls from zero to FFFFh, a terminal count is generated. If the channel is in auto-initialize mode, the Current Count register will be reloaded from the Base Count register after a terminal count is generated.</p> <p>For transfers to/from an 8-bit slave (channel's 0–3), the count register indicates the number of bytes to be transferred. For transfers to/from a 16-bit slave (channel's 5–7), the count register indicates the number of words to be transferred.</p> <p>The register is accessed in 8 bit quantities. The byte is pointed to by the current byte pointer flip/flop. Before accessing a count register, the byte pointer flip/flop should be cleared to ensure that the low byte is accessed first.</p> |

## 8.2.3 DMAMEM\_LP—DMA Memory Low Page Registers

|                |   |             |        |
|----------------|---|-------------|--------|
| I/O Address:   | Ch. #0 = 87h; Ch. #1 = 83h<br>Ch. #2 = 81h; Ch. #3 = 82h<br>Ch. #5 = 8Bh; Ch. #6 = 89h<br>Ch. #7 = 8Ah; | Attribute:  | R/W    |
| Default Value: | Undefined   | Size:       | 8 bits |
| Lockable:      | No  | Power Well: | Core   |

| Bit | Description   |
|-----|---|
| 7:0 | <p><b>DMA Low Page (ISA Address bits [23:16]).</b> This register works in conjunction with the DMA controller's Current Address Register to define the complete 24-bit address for the DMA channel. This register remains static throughout the DMA transfer.</p> |

## 8.2.4 DMACMD—DMA Command Register

I/O Address: Ch. #0–3 = 08h;  
Ch. #4–7 = D0h                      Attribute:              WO  
Default Value: Undefined              Size:                      8 bits  
Lockable: No                              Power Well:              Core

| Bit | Description   |
|-----|---|
| 7:5 | Reserved. Must be 0.  |
| 4   | <b>DMA Group Arbitration Priority.</b> Each channel group is individually assigned either fixed or rotating arbitration priority. At part reset, each group is initialized in fixed priority.<br>0 = Fixed priority to the channel group<br>1 = Rotating priority to the group. |
| 3   | Reserved. Must be 0   |
| 2   | <b>DMA Channel Group Enable.</b> Both channel groups are enabled following part reset.<br>1 = Disable. Disabling channel group 4–7 also disables channel group 0–3, which is cascaded through channel 4.<br>0 = Enable the DMA channel group.                                   |
| 1:0 | Reserved. Must be 0.  |

## 8.2.5 DMASTA—DMA Status Register

I/O Address: Ch. #0–3 = 08h;  
Ch. #4–7 = D0h                      Attribute:              RO  
Default Value: Undefined              Size:                      8 bits  
Lockable: No                              Power Well:              Core

| Bit | Description  |
|-----|--|
| 7:4 | <b>Channel Request Status.</b> When a valid DMA request is pending for a channel, the corresponding bit is set to 1. When a DMA request is not pending for a particular channel, the corresponding bit is set to 0. The source of the DREQ may be hardware or a software request. Note that channel 4 is the cascade channel, so the request status of channel 4 is a logical OR of the request status for channels 0 through 3.<br>4 = Channel 0<br>5 = Channel 1 (5)<br>6 = Channel 2 (6)<br>7 = Channel 3 (7) |
| 3:0 | <b>Channel Terminal Count Status.</b> When a channel reaches terminal count (TC), its status bit is set to 1. If TC has not been reached, the status bit is set to 0. Channel 4 is programmed for cascade, so the TC bit response for channel 4 is irrelevant:<br>0 = Channel 0<br>1 = Channel 1 (5)<br>2 = Channel 2 (6)<br>3 = Channel 3 (7)   |

## 8.2.6 DMA\_WRSMSK—DMA Write Single Mask Register

|                |                                   |             |        |
|----------------|-----------------------------------|-------------|--------|
| I/O Address:   | Ch. #0–3 = 0Ah;<br>Ch. #4–7 = D4h | Attribute:  | WO     |
| Default Value: | 0000 01xx                         | Size:       | 8 bits |
| Lockable:      | No                                | Power Well: | Core   |

| Bit | Description  |
|-----|--|
| 7:3 | Reserved. Must be 0.   |
| 2   | <b>Channel Mask Select.</b><br>1 = Disable DREQ for the selected channel.<br>0 = Enable DREQ for the selected channel. The channel is selected through bits [1:0]. Therefore, only one channel can be masked / unmasked at a time. |
| 1:0 | <b>DMA Channel Select.</b> These bits select the DMA Channel Mode Register to program.<br>00 = Channel 0 (4)<br>01 = Channel 1 (5)<br>10 = Channel 2 (6)<br>11 = Channel 3 (7)   |

## 8.2.7 DMACH\_MODE—DMA Channel Mode Register

|                |                                   |             |        |
|----------------|-----------------------------------|-------------|--------|
| I/O Address:   | Ch. #0–3 = 0Bh;<br>Ch. #4–7 = D6h | Attribute:  | WO     |
| Default Value: | 0000 00xx                         | Size:       | 8 bits |
| Lockable:      | No                                | Power Well: | Core   |

| Bit | Description  |
|-----|--|
| 7:6 | <b>DMA Transfer Mode.</b> Each DMA channel can be programmed in one of four different modes:<br>00 = Demand mode<br>01 = Single mode<br>10 = Reserved<br>11 = Cascade mode   |
| 5   | <b>Address Increment/Decrement Select.</b> This bit controls address increment/decrement during DMA transfers.<br>0 = Address increment. (default after part reset or Master Clear)<br>1 = address decrement.  |
| 4   | <b>Autoinitialize Enable.</b><br>1 = DMA restores the Base Address and Count registers to the current registers following a terminal count (TC).<br>0 = Autoinitialize feature is disabled and DMA transfers terminate on a terminal count. A part reset or Master Clear disables autoinitialization.  |
| 3:2 | <b>DMA Transfer Type.</b> These bits represent the direction of the DMA transfer. When the channel is programmed for cascade mode, (bits[7:6] = "11") the transfer type is irrelevant.<br>00 = Verify—No I/O or memory strobes generated<br>01 = Write—Data transferred from the I/O devices to memory<br>10 = Read—Data transferred from memory to the I/O device<br>11 = Illegal |
| 1:0 | <b>DMA Channel Select.</b> These bits select the DMA Channel Mode Register that will be written by bits [7:2].<br>00 = Channel 0 (4)<br>01 = Channel 1 (5)<br>10 = Channel 2 (6)<br>11 = Channel 3 (7)   |

## 8.2.8 DMA Clear Byte Pointer Register

I/O Address: Ch. #0–3 = 0Ch;  
Ch. #4–7 = D8h                      Attribute:              WO  
Default Value: xxxx xxxx              Size:                      8 bits  
Lockable: No                              Power Well:              Core

| Bit | Description  |
|-----|--|
| 7:0 | <b>Clear Byte Pointer.</b> No specific pattern. Command enabled with a write to the I/O port address. Writing to this register initializes the byte pointer flip/flop to a known state. It clears the internal latch used to address the upper or lower byte of the 16-bit Address and Word Count Registers. The latch is also cleared by part reset and by the Master Clear command. This command precedes the first access to a 16 bit DMA controller register. The first access to a 16 bit register will then access the significant byte, and the second access automatically accesses the most significant byte. |

## 8.2.9 DMA Master Clear Register

I/O Address: Ch. #0–3 = 0Dh;  
Ch. #4–7 = DAh                      Attribute:              WO  
Default Value: xxxx xxxx              Size:                      8 bits

| Bit | Description  |
|-----|--|
| 7:0 | <b>Master Clear.</b> No specific pattern. Enabled with a write to the port. This has the same effect as the hardware Reset. The Command, Status, Request, and Byte Pointer flip/flop registers are cleared and the Mask Register is set. |

## 8.2.10 DMA\_\_CLMSK—DMA Clear Mask Register

I/O Address: Ch. #0–3 = 0Eh;  
Ch. #4–7 = DCh                      Attribute:              WO  
Default Value: xxxx xxxx              Size:                      8 bits  
Lockable: No                              Power Well:              Core

| Bit | Description  |
|-----|--|
| 7:0 | <b>Clear Mask Register.</b> No specific pattern. Command enabled with a write to the port. |

## 8.2.11 DMA\_WRMSK—DMA Write All Mask Register

|                |                                   |             |        |
|----------------|-----------------------------------|-------------|--------|
| I/O Address:   | Ch. #0–3 = 0Fh;<br>Ch. #4–7 = DEh | Attribute:  | R/W    |
| Default Value: | 0000 1111                         | Size:       | 8 bits |
| Lockable:      | No                                | Power Well: | Core   |

| Bit | Description  |
|-----|--|
| 7:4 | Reserved. Must be 0.   |
| 3:0 | <p><b>Channel Mask Bits.</b> This register permits all four channels to be simultaneously enabled/disabled instead of enabling/disabling each channel individually, as is the case with the Mask Register - Write Single Mask Bit. In addition, this register has a read path to allow the status of the channel mask bits to be read. A channel's mask bit is automatically set to 1 when the Current Byte/Word Count Register reaches terminal count (unless the channel is in auto-initialization mode).</p> <p>Setting the bit(s) to a 1 disables the corresponding DREQ(s). Setting the bit(s) to a 0 enables the corresponding DREQ(s). Bits [3:0] are set to 1 upon part reset or Master Clear. When read, bits [3:0] indicate the DMA channel [3:0] ([7:4]) mask status.</p> <p>Bit 0 = Channel 0 (4)    1 = Masked, 0 = Not Masked<br/>                     Bit 1 = Channel 1 (5)    1 = Masked, 0 = Not Masked<br/>                     Bit 2 = Channel 2 (6)    1 = Masked, 0 = Not Masked<br/>                     Bit 3 = Channel 3 (7)    1 = Masked, 0 = Not Masked</p> <p><b>NOTE:</b> Disabling channel 4 also disables channels 0-3 due to the cascade of channel's 0-3 through channel 4.</p> |

## 8.3 Timer I/O Registers

| Port | Aliases | Register Name/Function                     | Default Value | Type |
|------|---------|--|---------------|------|
| 40h  | 50h     | Counter 0 Interval Time Status Byte Format | 0XXXXXXb      | RO   |
|      |         | Counter 0 Counter Access Port Register     | Undefined     | R/W  |
| 41h  | 51h     | Counter 1 Interval Time Status Byte Format | 0XXXXXXb      | RO   |
|      |         | Counter 1 Counter Access Port Register     | Undefined     | R/W  |
| 42h  | 52h     | Counter 2 Interval Time Status Byte Format | 0XXXXXXb      | RO   |
|      |         | Counter 2 Counter Access Port Register     | Undefined     | R/W  |
| 43h  | 53h     | Timer Control Word Register                | Undefined     | WO   |
|      |         | Timer Control Word Register Read Back      | XXXXXXX0b     | WO   |
|      |         | Counter Latch Command                      | X0h           | WO   |

### 8.3.1 TCW—Timer Control Word Register

I/O Address: 43h  
Attribute: WO

Default Value: All bits undefined  
Size: 8 bits

This register is programmed prior to any counter being accessed to specify counter modes. Following part reset, the control words for each register are undefined and each counter output is 0. Each timer must be programmed to bring it into a known state.

| Bit | Description   |
|-----|---|
| 7:6 | <b>Counter Select.</b> The Counter Selection bits select the counter the control word acts upon as shown below. The Read Back Command is selected when bits[7:6] are both 1.<br>00 = Counter 0 select<br>01 = Counter 1 select<br>10 = Counter 2 select<br>11 = Read Back Command   |
| 5:4 | <b>Read/Write Select.</b> These bits are the read/write control bits. The actual counter programming is done through the counter port (40h for counter 0, 41h for counter 1, and 42h for counter 2).<br>00 = Counter Latch Command<br>01 = Read/Write Least Significant Byte (LSB)<br>10 = Read/Write Most Significant Byte (MSB)<br>11 = Read/Write LSB then MSB                             |
| 3:1 | <b>Counter Mode Selection.</b> These bits select one of six possible modes of operation for the selected counter.<br>000 = Mode 0 Out signal on end of count (=0)<br>001 = Mode 1 Hardware retriggerable one-shot<br>x10 = Mode 2 Rate generator (divide by n counter)<br>x11 = Mode 3 Square wave output<br>100 = Mode 4 Software triggered strobe<br>101 = Mode 5 Hardware triggered strobe |
| 0   | <b>Binary/BCD Countdown Select.</b><br>0 = Binary countdown is used. The largest possible binary count is $2^{16}$<br>1 = Binary coded decimal (BCD) count is used. The largest possible BCD count is $10^4$  |

There are two special commands that can be issued to the counters through this register, the Read Back Command and the Counter Latch Command. When these commands are chosen, several bits within this register are redefined. These register formats are described below.

### 8.3.1.1 RDBK\_CMD—Read Back Command

The Read Back Command is used to determine the count value, programmed mode, and current states of the OUT pin and Null count flag of the selected counter or counters. Status and/or count may be latched in any or all of the counters by selecting the counter during the register write. The count and status remain latched until read, and further latch commands are ignored until the count is read. Both count and status of the selected counters may be latched simultaneously by setting both bit 5 and bit 4 to 0. If both are latched, the first read operation from that counter returns the latched status. The next one or two reads, depending on whether the counter is programmed for one or two byte counts, returns the latched count. Subsequent reads return an unlatched count.

| Bit | Description   |
|-----|---|
| 7:6 | <b>Read Back Command.</b> Must be "11" to select the Read Back Command  |
| 5   | <b>Latch Count of Selected Counters.</b><br>0 = Current count value of the selected counters will be latched<br>1 = Current count will not be latched |
| 4   | <b>Latch Status of Selected Counters.</b><br>0 = Status of the selected counters will be latched<br>1 = Status will not be latched                    |
| 3   | <b>Counter 2 Select.</b><br>1 = Counter 2 count and/or status will be latched   |
| 2   | <b>Counter 1 Select.</b><br>1 = Counter 1 count and/or status will be latched   |
| 1   | <b>Counter 0 Select.</b><br>1 = Counter 0 count and/or status will be latched.  |
| 0   | Reserved. Must be 0.  |

### 8.3.1.2 LTCH\_CMD—Counter Latch Command

The Counter Latch Command latches the current count value. This command is used to insure that the count read from the counter is accurate. The count value is then read from each counter's count register through the Counter Ports Access Ports Register (40h for counter 0, 41h for counter 1, and 42h for counter 2). The count must be read according to the programmed format, i.e. if the counter is programmed for two byte counts, two bytes must be read. The two bytes do not have to be read one right after the other (read, write, or programming operations for other counters may be inserted between the reads). If a counter is latched once and then latched again before the count is read, the second Counter Latch Command is ignored.

| Bit | Description   |
|-----|---|
| 7:6 | <b>Counter Selection.</b> These bits select the counter for latching. If "11" is written, then the write is interpreted as a read back command.<br>00 = Counter 0<br>01 = Counter 1<br>10 = Counter 2 |
| 5:4 | <b>Counter Latch Command.</b><br>00 = Selects the Counter Latch Command.  |
| 3:0 | Reserved. Must be 0.  |

### 8.3.2 SBYTE\_FMT—Interval Timer Status Byte Format Register

I/O Address: Counter 0 = 40h,  
Counter 1 = 41h, Attribute: RO  
Counter 2 = 42h Size: 8 bits per counter  
Default Value: Bits[6:0] undefined, Bit 7=0

Each counter's status byte can be read following a Read Back Command. If latch status is chosen (bit 4=0, Read Back Command) as a read back option for a given counter, the next read from the counter's Counter Access Ports Register (40h for counter 0, 41h for counter 1, and 42h for counter 2) returns the status byte. The status byte returns the following:

| Bit | Description   |
|-----|---|
| 7   | <b>Counter OUT Pin State.</b><br>1 =OUT pin of the counter is also a 1.<br>0 =OUT pin of the counter is also a 0.   |
| 6   | <b>Count Register Status.</b> This bit indicates when the last count written to the Count Register (CR) has been loaded into the counting element (CE). The exact time this happens depends on the counter mode, but until the count is loaded into the counting element (CE), the count value will be incorrect.<br>0 = Count has been transferred from CR to CE and is available for reading.<br>1 = Null Count. Count has not been transferred from CR to CE and is not yet available for reading. |
| 5:4 | <b>Read/Write Selection Status.</b> These reflect the read/write selection made through bits[5:4] of the control register. The binary codes returned during the status read match the codes used to program the counter read/write selection.<br>00 = Counter Latch Command<br>01 = Read/Write Least Significant Byte (LSB)<br>10 = Read/Write Most Significant Byte (MSB)<br>11 = Read/Write LSB then MSB  |



| Bit | Description   |
|-----|---|
| 3:1 | <b>Mode Selection Status.</b> These bits return the counter mode programming. The binary code returned matches the code used to program the counter mode, as listed under the bit function above.<br>000 = Mode 0 Out signal on end of count (=0)<br>001 = Mode 1 Hardware retriggerable one-shot<br>x10 = Mode 2 Rate generator (divide by n counter)<br>x11 = Mode 3 Square wave output<br>100 = Mode 4 Software triggered strobe<br>101 = Mode 5 Hardware triggered strobe |
| 0   | <b>Countdown Type Status.</b> This bit reflects the current countdown type; ether 0 for binary countdown or a 1 for binary coded decimal (BCD) countdown.   |

### 8.3.3 Counter Access Ports Register

|                |   |            |       |
|----------------|---|------------|-------|
| I/O Address:   | Counter 0–40h,<br>Counter 1–41h,<br>Counter 2–42h | Attribute: | R/W   |
| Default Value: | All bits undefined                                | Size:      | 8 bit |

| Bit | Description  |
|-----|--|
| 7:0 | <b>Counter Port.</b> Each counter port address is used to program the 16-bit Count Register. The order of programming, either LSB only, MSB only, or LSB then MSB, is defined with the Interval Counter Control Register at port 43h. The counter port is also used to read the current count from the Count Register, and return the status of the counter programming following a Read Back Command. |

## 8.4 8259 Interrupt Controller (PIC) Registers

### 8.4.1 Interrupt Controller I/O MAP

The interrupt controller registers are located at 20h and 21h for the master controller (IRQ0:7), and at A0h and A1h for the slave controller (IRQ8:13). These registers have multiple functions, depending upon the data written to them. Below is a description of the different register possibilities for each address.

**Table 8-3. PIC Registers**

| Port | Aliases                                 | Register Name/Function                    | Default Value | Type |
|------|---|---|---------------|------|
| 20h  | 24h, 28h,<br>2Ch, 30h,<br>34h, 38h, 3Ch | Master PIC ICW1 Init. Cmd Word 1 Register | Undefined     | WO   |
|      |   | Master PIC OCW2 Op Ctrl Word 2 Register   | 001XXXXXb     | WO   |
|      |   | Master PIC OCW3 Op Ctrl Word 3 Register   | X01XXX10b     | R/W  |
| 21h  | 25h, 29h,<br>2Dh, 31h,<br>35h, 39h, 3Dh | Master PIC ICW2 Init. Cmd Word 2 Register | Undefined     | WO   |
|      |   | Master PIC ICW3 Init. Cmd Word 3 Register | Undefined     | WO   |
|      |   | Master PIC ICW4 Init. Cmd Word 4 Register | 01h           | WO   |
|      |   | Master PIC OCW1 Op Ctrl Word 1 Register   | 00h           | R/W  |
| A0h  | A4h, A8h,<br>ACh, B0h,<br>B4h, B8h, BCh | Slave PIC ICW1 Init. Cmd Word 1 Register  | Undefined     | WO   |
|      |   | Slave PIC OCW2 Op Ctrl Word 2 Register    | 001XXXXXb     | WO   |
|      |   | Slave PIC OCW3 Op Ctrl Word 3 Register    | X01XXX10b     | R/W  |
| A1h  | A5h, A9h,<br>ADh, B1h,<br>B5h, B9h, BDh | Slave PIC ICW2 Init. Cmd Word 2 Register  | Undefined     | WO   |
|      |   | Slave PIC ICW3 Init. Cmd Word 3 Register  | Undefined     | WO   |
|      |   | Slave PIC ICW4 Init. Cmd Word 4 Register  | 01h           | WO   |
|      |   | Slave PIC OCW1 Op Ctrl Word 1 Register    | 00h           | R/W  |
| 4D0h | -                                       | Master PIC Edge/Level Triggered Register  | 00h           | R/W  |
| 4D1h | -                                       | Slave PIC Edge/Level Triggered Register   | 00h           | R/W  |

## 8.4.2 ICW1—Initialization Command Word 1 Register

Offset Address:   Master Controller—020h           Attribute:            WO  
                       Slave Controller—0A0h         Size:                8 bit /controller  
 Default Value:    All bits undefined

A write to Initialization Command Word 1 starts the interrupt controller initialization sequence, during which the following occurs:

1. The Interrupt Mask register is cleared.
2. IRQ7 input is assigned priority 7.
3. The slave mode address is set to 7.
4. Special Mask Mode is cleared and Status Read is set to IRR.

Once this write occurs, the controller expects writes to ICW2, ICW3, and ICW4 to complete the initialization sequence.

| Bit | Description   |
|-----|---|
| 7:5 | <b>ICW/OCW select.</b> These bits are MCS-85 specific, and not needed.<br>000 = Should be programmed to "000"             |
| 4   | <b>ICW/OCW select.</b><br>1 = This bit must be a 1 to select ICW1 and enable the ICW2, ICW3, and ICW4 sequence.           |
| 3   | <b>Edge/Level Bank Select (LTIM).</b> Disabled. Replaced by the edge/level triggered control registers (ELCR).            |
| 2   | ADI.<br>0 = Ignored for the ICH. Should be programmed to '0'.   |
| 1   | <b>Single or Cascade (SNGL).</b><br>0 = Must be programmed to a 0 to indicate two controllers operating in cascade mode.  |
| 0   | <b>ICW4 Write Required (IC4).</b><br>1 = This bit must be programmed to a 1 to indicate that ICW4 needs to be programmed. |

### 8.4.3 ICW2—Initialization Command Word 2 Register

|                 |   |            |                   |
|-----------------|---|------------|-------------------|
| Offset Address: | Master Controller—021h<br>Slave Controller—0A1h | Attribute: | WO                |
| Default Value:  | All bits undefined                              | Size:      | 8 bit /controller |

ICW2 is used to initialize the interrupt controller with the five most significant bits of the interrupt vector address. The value programmed for bits[7:3] is used by the processor to define the base address in the interrupt vector table for the interrupt routines associated with each IRQ on the controller. Typical ISA ICW2 values are 08h for the master controller and 70h for the slave controller.

| Bit  | Description  |                 |                  |                 |     |      |      |     |      |      |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |
|------|--|-----------------|------------------|-----------------|-----|------|------|-----|------|------|-----|------|-------|-----|------|-------|-----|------|-------|-----|------|-------|-----|------|-------|-----|------|-------|
| 7:3  | <b>Interrupt Vector Base Address.</b> Bits [7:3] define the base address in the interrupt vector table for the interrupt routines associated with each interrupt request level input.  |                 |                  |                 |     |      |      |     |      |      |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |
| 2:0  | <p><b>Interrupt Request Level.</b> When writing ICW2, these bits should all be 0. During an interrupt acknowledge cycle, these bits are programmed by the interrupt controller with the interrupt to be serviced. This is combined with bits [7:3] to form the interrupt vector driven onto the data bus during the second INTA# cycle. The code is a three bit binary code:</p> <table border="1"> <thead> <tr> <th>Code</th> <th>Master Interrupt</th> <th>Slave Interrupt</th> </tr> </thead> <tbody> <tr> <td>000</td> <td>IRQ0</td> <td>IRQ8</td> </tr> <tr> <td>001</td> <td>IRQ1</td> <td>IRQ9</td> </tr> <tr> <td>010</td> <td>IRQ2</td> <td>IRQ10</td> </tr> <tr> <td>011</td> <td>IRQ3</td> <td>IRQ11</td> </tr> <tr> <td>100</td> <td>IRQ4</td> <td>IRQ12</td> </tr> <tr> <td>101</td> <td>IRQ5</td> <td>IRQ13</td> </tr> <tr> <td>110</td> <td>IRQ6</td> <td>IRQ14</td> </tr> <tr> <td>111</td> <td>IRQ7</td> <td>IRQ15</td> </tr> </tbody> </table> | Code            | Master Interrupt | Slave Interrupt | 000 | IRQ0 | IRQ8 | 001 | IRQ1 | IRQ9 | 010 | IRQ2 | IRQ10 | 011 | IRQ3 | IRQ11 | 100 | IRQ4 | IRQ12 | 101 | IRQ5 | IRQ13 | 110 | IRQ6 | IRQ14 | 111 | IRQ7 | IRQ15 |
| Code | Master Interrupt   | Slave Interrupt |                  |                 |     |      |      |     |      |      |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |
| 000  | IRQ0   | IRQ8            |                  |                 |     |      |      |     |      |      |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |
| 001  | IRQ1   | IRQ9            |                  |                 |     |      |      |     |      |      |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |
| 010  | IRQ2   | IRQ10           |                  |                 |     |      |      |     |      |      |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |
| 011  | IRQ3   | IRQ11           |                  |                 |     |      |      |     |      |      |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |
| 100  | IRQ4   | IRQ12           |                  |                 |     |      |      |     |      |      |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |
| 101  | IRQ5   | IRQ13           |                  |                 |     |      |      |     |      |      |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |
| 110  | IRQ6   | IRQ14           |                  |                 |     |      |      |     |      |      |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |
| 111  | IRQ7   | IRQ15           |                  |                 |     |      |      |     |      |      |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |     |      |       |

### 8.4.4 ICW3—Master Controller Initialization Command Word 3 Register

|                 |                    |            |        |
|-----------------|--------------------|------------|--------|
| Offset Address: | 21h                | Attribute: | WO     |
| Default Value:  | All bits undefined | Size:      | 8 bits |

| Bit | Description  |
|-----|--|
| 7:3 | 0 = These bits must be programmed to zero.   |
| 2   | <p><b>Cascaded Interrupt Controller IRQ Connection.</b> This bit indicates that the slave controller is cascaded on IRQ2. When IRQ8#–IRQ15 is asserted, it goes through the slave controller's priority resolver. The slave controller's INTR output onto IRQ2. IRQ2 then goes through the master controller's priority solver. If it wins, the INTR signal is asserted to the processor, and the returning interrupt acknowledge returns the interrupt vector for the slave controller.</p> <p>1 = This bit must always be programmed to a 1.</p> |
| 1:0 | 0 = These bits must be programmed to zero.   |

### 8.4.5 ICW3—Slave Controller Initialization Command Word 3 Register

Offset Address: A1h Attribute: WO  
 Default Value: All bits undefined Size: 8 bits

| Bit | Description  |
|-----|--|
| 7:3 | Reserved. Must be 0.   |
| 2:0 | <b>Slave Identification Code.</b> These bits are compared against the slave identification code broadcast by the master controller from the trailing edge of the first internal INTA# pulse to the trailing edge of the second internal INTA# pulse. These bits must be programmed to 02h to match the code broadcast by the master controller. When 02h is broadcast by the master controller during the INTA# sequence, the slave controller assumes responsibility for broadcasting the interrupt vector. |

### 8.4.6 ICW4—Initialization Command Word 4 Register

Offset Address: Master Controller–021h Attribute: WO  
 Slave Controller–0A1h Size: 8 bits

| Bit | Description  |
|-----|--|
| 7:5 | Reserved. Must be 0.   |
| 4   | <b>Special Fully Nested Mode (SFNM).</b> Should normally be disabled by writing a 0 to this bit. If SFNM=1, the special fully nested mode is programmed.   |
| 3   | <b>Buffered Mode (BUF).</b> Must be programmed to 0 for the ICH. This is non-buffered mode.  |
| 2   | <b>Master/Slave in Buffered Mode.</b> Not used. Should always be programmed to 0.  |
| 1   | <b>Automatic End of Interrupt (AEOI).</b> This bit should normally be programmed to 0. This is the normal end of interrupt. If this bit is 1, the automatic end of interrupt mode is programmed. AEOI is discussed in Section 16.10.2. |
| 0   | <b>Microprocessor Mode.</b> This bit must be programmed to 1 to indicate that the controller is operating in an Intel Architecture-based system. Programming this bit to 0 will result in improper controller operation.               |

### 8.4.7 OCW1—Operational Control Word 1 (Interrupt Mask) Register

Offset Address: Master Controller–021h Attribute: R/W  
 Slave Controller–0A1h Size: 8 bits  
 Default Value: 00h

| Bit | Description   |
|-----|---|
| 7:0 | <b>Interrupt Request Mask.</b> When a 1 is written to any bit in this register, the corresponding IRQ line is masked. When a 0 is written to any bit in this register, the corresponding IRQ mask bit is cleared, and interrupt requests will again be accepted by the controller. Masking IRQ2 on the master controller will also mask the interrupt requests from the slave controller. |

### 8.4.8 OCW2—Operational Control Word 2 Register

Offset Address: Master Controller—020h      Attribute: WO  
 Slave Controller—0A0h      Size: 8 bits  
 Default Value: Bit[4:0]=undefined, Bit[7:5]=001

Following a part reset or ICW initialization, the controller enters the fully nested mode of operation. Non-specific EOI without rotation is the default. Both rotation mode and specific EOI mode are disabled following initialization.

| Bit  | Description  |      |                 |      |                 |     |        |     |         |     |        |     |         |     |         |     |         |     |         |     |         |
|------|--|------|-----------------|------|-----------------|-----|--------|-----|---------|-----|--------|-----|---------|-----|---------|-----|---------|-----|---------|-----|---------|
| 7:5  | <p><b>Rotate and EOI Codes (R, SL, EOI).</b> These three bits control the Rotate and End of Interrupt modes and combinations of the two. A chart of these combinations is listed above under the bit definition.</p> <p>000 = Rotate in Auto EOI Mode (Clear)<br/>           001 = Non-specific EOI command<br/>           010 = No Operation<br/>           011 = Specific EOI Command<br/>           100 = Rotate in Auto EOI Mode (Set)<br/>           101 = Rotate on Non-Specific EOI Command<br/>           110 = *Set Priority Command<br/>           111 = *Rotate on Specific EOI Command<br/>           *L0–L2 Are Used</p>  |      |                 |      |                 |     |        |     |         |     |        |     |         |     |         |     |         |     |         |     |         |
| 4:3  | <p><b>OCW2 Select.</b> When selecting OCW2, bits 4:3 = "00"</p>  |      |                 |      |                 |     |        |     |         |     |        |     |         |     |         |     |         |     |         |     |         |
| 2:0  | <p><b>Interrupt Level Select (L2, L1, L0).</b> L2, L1, and L0 determine the interrupt level acted upon when the SL bit is active. A simple binary code, outlined above, selects the channel for the command to act upon. When the SL bit is inactive, these bits do not have a defined function; programming L2, L1 and L0 to 0 is sufficient in this case.</p> <table border="1"> <thead> <tr> <th>Bits</th> <th>Interrupt Level</th> <th>Bits</th> <th>Interrupt Level</th> </tr> </thead> <tbody> <tr> <td>000</td> <td>IRQ0/8</td> <td>100</td> <td>IRQ4/12</td> </tr> <tr> <td>001</td> <td>IRQ1/9</td> <td>101</td> <td>IRQ5/13</td> </tr> <tr> <td>010</td> <td>IRQ2/10</td> <td>110</td> <td>IRQ6/14</td> </tr> <tr> <td>011</td> <td>IRQ3/11</td> <td>111</td> <td>IRQ7/15</td> </tr> </tbody> </table> | Bits | Interrupt Level | Bits | Interrupt Level | 000 | IRQ0/8 | 100 | IRQ4/12 | 001 | IRQ1/9 | 101 | IRQ5/13 | 010 | IRQ2/10 | 110 | IRQ6/14 | 011 | IRQ3/11 | 111 | IRQ7/15 |
| Bits | Interrupt Level  | Bits | Interrupt Level |      |                 |     |        |     |         |     |        |     |         |     |         |     |         |     |         |     |         |
| 000  | IRQ0/8   | 100  | IRQ4/12         |      |                 |     |        |     |         |     |        |     |         |     |         |     |         |     |         |     |         |
| 001  | IRQ1/9   | 101  | IRQ5/13         |      |                 |     |        |     |         |     |        |     |         |     |         |     |         |     |         |     |         |
| 010  | IRQ2/10  | 110  | IRQ6/14         |      |                 |     |        |     |         |     |        |     |         |     |         |     |         |     |         |     |         |
| 011  | IRQ3/11  | 111  | IRQ7/15         |      |                 |     |        |     |         |     |        |     |         |     |         |     |         |     |         |     |         |

### 8.4.9 OCW3—Operational Control Word 3 Register

Offset Address: Master Controller—020h      Attribute: WO  
                   Slave Controller—0A0h      Size: 8 bits  
 Default Value: Bit[6,0]=0, Bit[7,4:2]=undefined,  
                   Bit[5,1]=1

| Bit | Description  |
|-----|--|
| 7   | Reserved. Must be 0.   |
| 6   | <b>Special Mask Mode (SMM).</b><br>1 = The Special Mask Mode can be used by an interrupt service routine to dynamically alter the system priority structure while the routine is executing, through selective enabling/disabling of the other channel's mask bits. Bit 6, the ESMM bit, must be set for this bit to have any meaning.  |
| 5   | <b>Enable Special Mask Mode (ESMM).</b><br>1 = Enable the SMM bit to set or reset the Special Mask Mode.<br>0 = the SMM bit becomes a "don't care".  |
| 4:3 | <b>OCW3 Select.</b> When selecting OCW3, bits 4:3 = "01"   |
| 2   | <b>Poll Mode Command.</b><br>0 = Disable. Poll Command is not issued.<br>1 = Enable. The next I/O read to the interrupt controller is treated as an interrupt acknowledge cycle. An encoded byte is driven onto the data bus, representing the highest priority level requesting service.  |
| 1:0 | <b>Register Read Command.</b> These bits provide control for reading the In-Service Register (ISR) and the Interrupt Request Register (IRR). When bit 1=0, bit 0 will not affect the register read selection. When bit 1=1, bit 0 selects the register status returned following an OCW3 read. If bit 0=0, the IRR will be read. If bit 0=1, the ISR will be read. Following ICW initialization, the default OCW3 port address read will be "read IRR". To retain the current selection (read ISR or read IRR), always write a 0 to bit 1 when programming this register. The selected register can be read repeatedly without reprogramming OCW3. To select a new status register, OCW3 must be reprogrammed prior to attempting the read.<br>00 = No Action<br>01 = No Action<br>10 = Read IRQ Register<br>11 = Read IS Register |

### 8.4.10 ELCR1—Master Controller Edge/Level Triggered Register

Offset Address: 4D0h                                      Attribute: R/W  
Default Value: 00h                                        Size: 8 bits

In edge mode, (bit[x] = 0), the interrupt is recognized by a low to high transition. In level mode (bit[x] = 1), the interrupt is recognized by a high level. The cascade channel, IRQ2, the heart beat timer (IRQ0), and the keyboard controller (IRQ1), cannot be put into level mode.

| Bit | Description                    |
|-----|--------------------------------|
| 7   | IRQ7 ECL. 0 = Edge. 1 = Level. |
| 6   | IRQ6 ECL. 0 = Edge. 1 = Level. |
| 5   | IRQ5 ECL. 0 = Edge. 1 = Level. |
| 4   | IRQ4 ECL. 0 = Edge. 1 = Level. |
| 3   | IRQ3 ECL. 0 = Edge. 1 = Level. |
| 2:0 | Reserved. Must be 0.           |

### 8.4.11 ELCR2—Slave Controller Edge/Level Triggered Register

Offset Address: 4D1h                                      Attribute: R/W  
Default Value: 00h                                        Size: 8 bits

In edge mode, (bit[x] = 0), the interrupt is recognized by a low to high transition. In level mode (bit[x] = 1), the interrupt is recognized by a high level. The real time clock, IRQ8#, and the floating point error interrupt, IRQ13, cannot be programmed for level mode.

| Bit | Description                     |
|-----|---------------------------------|
| 7   | IRQ15 ECL. 0 = Edge. 1 = Level. |
| 6   | IRQ14 ECL. 0 = Edge. 1 = Level. |
| 5   | Reserved. Must be 0.            |
| 4   | IRQ12 ECL. 0 = Edge. 1 = Level. |
| 3   | IRQ11 ECL. 0 = Edge. 1 = Level. |
| 2   | IRQ10 ECL. 0 = Edge. 1 = Level. |
| 1   | IRQ9 ECL. 0 = Edge. 1 = Level.  |
| 0   | Reserved. Must be 0.            |



## 8.5 Advanced Interrupt Controller (APIC)

### 8.5.1 APIC Register Map

The APIC is accessed via an indirect addressing scheme. Two registers are visible by software for manipulation of most of the APIC registers. These registers are mapped into memory space. The registers are shown in Table 8-4.

Table 8-4. APIC Direct Registers

| Address    | Register                   | Size    | Type |
|------------|----------------------------|---------|------|
| FEC0_0000h | Index Register             | 8 bits  | R/W  |
| FEC0_0010h | Data Register              | 32 bits | R/W  |
| FEC0_0020h | IRQ Pin Assertion Register | 8 bits  | WO   |
| FEC0_0040h | EOI Register               | 8 bits  | WO   |

Table 8-5 lists the registers which can be accessed within the APIC via the Index Register. When accessing these registers, accesses must be done a DWord at a time. For example, software should never access byte 2 from the Data register before accessing bytes 0 and 1. The hardware will not attempt to recover from a bad programming model in this case.

Table 8-5. APIC Indirect Registers

| Index | Register             | Size    | Type |
|-------|----------------------|---------|------|
| 00    | ID                   | 32 bits | R/W  |
| 01    | Version              | 32 bits | RO   |
| 02    | Arbitration ID       | 32 bits | RO   |
| 03–0F | Reserved             |         | RO   |
| 10–11 | Redirection Table 0  | 64 bits | R/W  |
| 12–13 | Redirection Table 1  | 64 bits | R/W  |
| 3E–3F | Redirection Table 23 | 64 bits | R/W  |
| 40–FF | Reserved             |         | RO   |

### 8.5.2 IND—Index Register

Memory Address: FEC0\_0000h      Attribute: R/W  
 Default Value: 00h                  Size: 8 bits

The Index Register will select which APIC indirect register to be manipulated by software. The selector values for the indirect registers are listed in Table 8-5. Software will program this register to select the desired APIC internal register

| Bit | Description   |
|-----|---|
| 7:0 | <b>APIC Index.</b> This is an 8 bit pointer into the I/O APIC register table. |

### 8.5.3 DAT—Data Register

Memory Address    FEC0\_0010h                    Attribute:            R/W  
 Default Value:    00000000h                    Size:                32 bits

This is a 32 bit register specifying the data to be read or written to the register pointed to by the Index register. This register can be accessed in byte quantities.

| Bit | Description  |
|-----|--|
| 7:0 | <b>APIC Data.</b> This is a 32 bit register for the data to be read or written to the APIC indirect register pointed to by the Index register. |

### 8.5.4 ID—Identification Register

Index Offset:        00h    Attribute:            R/W  
 Default Value:       00000000h                    Size:                32 bits

The APIC ID serves as a physical name of the APIC. The APIC bus arbitration ID for the APIC is derived from its I/O APIC ID. This register is reset to zero on power up reset

| Bit   | Description   |
|-------|---|
| 31:28 | Reserved.   |
| 27:24 | <b>APIC ID.</b> Software must program this value before using the APIC. |
| 23:0  | Reserved.   |

### 8.5.5 VER—Version Register

Index Offset:        01h    Attribute:            RO  
 Default Value:       00170011h                    Size:                32 bits

Each I/O APIC contains a hardwired Version Register that identifies different implementation of APIC and their versions. The maximum redirection entry information also is in this register, to let software know how many interrupt are supported by this APIC.

| Bit   | Description  |
|-------|--|
| 31:24 | Reserved.  |
| 23:16 | <b>Maximum Redirection Entries.</b> This is the entry number (0 being the lowest entry) of the highest entry in the redirection table. It is equal to the number of interrupt input pins minus one and is in the range 0 through 239. This field is hardwired and is read-only. In the ICH this field is hardwired to 17h to indicate 24 interrupts. |
| 15    | <b>PRQ.</b> This bit is set to 1 to indicate that this version of the I/O APIC implements the IRQ Assertion register and allows PCI devices to write to it to cause interrupts.  |
| 14 :8 | Reserved.  |
| 7:0   | <b>Version.</b> This is a version number that identifies the implementation version. This field is hardwired and is read only.   |

## 8.5.6 ARBID—Arbitration ID Register

|                |           |            |         |
|----------------|-----------|------------|---------|
| Index Offset:  | 02h       | Attribute: | RO      |
| Default Value: | 00000000h | Size:      | 32 bits |

This register contains the bus arbitration priority for the APIC. This register is loaded whenever the APIC ID register is loaded. A rotating priority scheme is used for APIC bus arbitration. The winner of the arbitration becomes the lowest priority agent and assumes an arbitration ID of 0.

| Bit   | Description  |
|-------|--|
| 31:28 | Reserved.  |
| 27:24 | <b>I/O APIC Identification.</b> This 4 bit field contains the I/O APIC Arbitration ID. |
| 23:0  | Reserved.  |

## 8.5.7 Redirection Table

|                |  |            |  |
|----------------|--|------------|--|
| Index Offset:  | 10h–11h (vector 0) through<br>3E–3Fh (vector 23)         | Attribute: | R/W  |
| Default Value: | Bit 16='1', Bits[15:12]='0'.<br>All other bits undefined | Size:      | 64 bits each, (accessed as<br>two 32 bit quantities) |

The Redirection Table has a dedicated entry for each interrupt input pin. The information in the Redirection Table is used to translate the interrupt manifestation on the corresponding interrupt pin into an APIC message.

The APIC will respond to an edge triggered interrupt as long as the interrupt is held until after the acknowledge cycle has begun. Once the interrupt is detected, a delivery status bit internally to the I/O APIC is set. The state machine will step ahead and wait for an acknowledgment from the APIC bus unit that the interrupt message was sent over the APIC bus. Only then will the I/O APIC be able to recognize a new edge on that interrupt pin. That new edge will only result in a new invocation of the handler if its acceptance by the destination APIC causes the Interrupt Request Register bit to go from 0 to 1. (In other words, if the interrupt was not already pending at the destination.)

| Bit   | Description   |
|-------|---|
| 63:56 | <b>Destination.</b> If bit 11 of this entry is 0 [Physical], then bits [59:56] specifies an APIC ID. If bit 11 of this entry is 1 [Logical], then bits [63:56] specify the logical destination address of a set of processors.  |
| 55:17 | Reserved.   |
| 16    | <b>Mask.</b><br>0 = Not masked: An edge or level on this interrupt pin results in the delivery of the interrupt to the destination.<br>1 = Masked: Interrupts are not delivered nor held pending. Setting this bit after the interrupt is accepted by a local APIC has no effect on that interrupt. This behavior is identical to the device withdrawing the interrupt before it is posted to the processor. It is software's responsibility to deal with the case where the mask bit is set after the interrupt message has been accepted by a local APIC unit but before the interrupt is dispensed to the processor. |
| 15    | <b>Trigger Mode.</b> This field indicates the type of signal on the interrupt pin that triggers an interrupt.<br>0 = Indicates edge sensitive<br>1 = Indicates level sensitive.   |

| Bit  | Description  |
|------|--|
| 14   | <p><b>Remote IRR.</b> This bit is used for level triggered interrupts; its meaning is undefined for edge triggered interrupts.</p> <p>1 = For level triggered interrupts, this bit is set when Local APIC/s accept the level interrupt sent by the I/O APIC.</p> <p>0 = Remote IRR bit is reset when an EOI message is received from a local APIC.</p>   |
| 13   | <p><b>Interrupt Input Pin Polarity.</b> This bit specifies the polarity of each interrupt signal connected to the interrupt pins. A value of 0 means the signal is active high. A value of 1 means the signal is active low.</p>   |
| 12   | <p><b>Delivery Status—RO.</b> This field contains the current status of the delivery of this interrupt. Writes to this bit have no effect.</p> <p>0 = Idle. No activity for this interrupt</p> <p>1 = Pending. Interrupt has been injected, but delivery is held up due to the APIC bus being busy or the inability of the receiving APIC unit to accept the interrupt at this time.</p>   |
| 11   | <p><b>Destination Mode.</b> This field determines the interpretation of the Destination field.</p> <p>0 = Physical. Destination APIC ID is identified by bits [59:56].</p> <p>1 = Logical. Destinations are identified by matching bit [63:56] with the Logical Destination in the Destination Format Register and Logical Destination Register in each Local APIC.</p>  |
| 10:8 | <p><b>Delivery Mode.</b> This field specifies how the APICs listed in the destination field should act upon reception of this signal. Certain Delivery Modes will only operate as intended when used in conjunction with a specific trigger mode. These encodings are:</p> <p>000 = Fixed. Deliver the signal on the INTR signal of all processor cores listed in the destination. Trigger Mode can be edge or level.</p> <p>001 = Lowest Priority. Deliver the signal on the INTR signal of the processor core that is executing at the lowest priority among all the processors listed in the specified destination. Trigger Mode can be edge or level.</p> <p>010 = SMI (System Management Interrupt). Requires the interrupt to be programmed as edge triggered. The vector information is ignored but must be programmed to all zeroes for future compatibility.</p> <p>011 = Reserved</p> <p>100 = NMI. Deliver the signal on the NMI signal of all processor cores listed in the destination. Vector information is ignored. NMI is treated as an edge triggered interrupt even if it is programmed as level triggered. For proper operation this redirection table entry must be programmed to edge triggered. The NMI delivery mode does not set the RIRR bit. Once the interrupt is detected, it will be sent over the APIC bus.</p> <p>If the redirection table is incorrectly set to level, the loop count will continue counting through the redirection table addresses. Once the count for the NMI pin is reached again, the interrupt will be sent over the APIC bus again.</p> <p>101 = INIT. Deliver the signal to all processor cores listed in the destination by asserting the INIT</p> |
| 7:0  | <p><b>Vector.</b> This field contains the interrupt vector for this interrupt. Values range between 10h and FEh.</p>   |

## 8.5.8 IRQPA—IRQ Pin Assertion Register

|                |            |            |         |
|----------------|------------|------------|---------|
| Memory Address | FEC0_0020h | Attribute: | WO      |
| Default Value: | N/A        | Size:      | 32 bits |

The IRQ Pin Assertion Register is present to provide a mechanism to scale the number of interrupt inputs into the I/O APIC without increasing the number of dedicated input pins. When a device that supports this interrupt assertion protocol requires interrupt service, that device will issue a write to this register. Bits 4:0 written to this register contain the IRQ number for this interrupt. The only valid values are 0–23. Bits 31:5 are ignored. See Section below for more details on how PCI devices will use this field.

To provide for future expansion, peripherals should always write a value of 0 for Bits 31:5.

| Bit  | Description   |
|------|---|
| 31:5 | Reserved. Bits 31:5 are ignored.  |
| 4:0  | <b>IRQ Number.</b> Bits 4:0 written to this register contain the IRQ number for this interrupt. The only valid values are 0–23. |

## 8.5.9 EOIR—EOI Register

|                |            |            |         |
|----------------|------------|------------|---------|
| Memory Address | FEC0_0040h | Attribute: | WO      |
| Default Value: | N/A        | Size:      | 32 bits |

The EOI register is present to provide a mechanism to maintain the level triggered semantics for level-triggered interrupts issued on the parallel bus.

When a write is issued to this register, the I/O APIC will check the lower 8 bits written to this register, and compare it with the vector field for each entry in the I/O Redirection Table. When a match is found, the Remote\_IRR bit for that I/O Redirection Entry will be cleared.

Note: This is similar to what already occurs when the APIC sees the EIO message on the serial bus. Note that if multiple I/O Redirection entries, for any reason, assign the same vector for more than one interrupt input, each of those entries will have the Remote\_IRR bit reset to '0'. The interrupt which was prematurely reset will not be lost because if its input remained active when the Remote\_IRR bit is cleared, the interrupt will be reissued and serviced at a later time. Note: Only bits 7:0 are actually used. Bits 31:8 are ignored by the ICH.

**Note:** To provide for future expansion, the processor should always write a value of 0 to Bits 31:8.

| Bit  | Description  |
|------|--|
| 31:8 | Reserved. To provide for future expansion, the processor should always write a value of 0 to Bits 31:8.  |
| 7:0  | <b>Redirection Entry Clear.</b> When a write is issued to this register, the I/O APIC will check this field, and compare it with the vector field for each entry in the I/O Redirection Table. When a match is found, the Remote_IRR bit for that I/O Redirection Entry will be cleared. |

## 8.6 Real Time Clock Registers

### 8.6.1 I/O Register Address Map

The RTC internal registers and RAM are organized as two banks of 128 bytes each, called the standard and extended banks. The first 14 bytes of the standard bank contain the RTC time and date information along with four registers (A–D) that are used for configuration of the RTC. The extended bank contains a full 128 bytes of battery backed SRAM, and will be accessible even when the RTC module is disabled (via the RTC configuration register). Registers A–D do not physically exist in the RAM.

All data movement between the host processor and the real-time clock is done through registers mapped to the standard I/O space. The register map appears in Table 8-6.

**Table 8-6. RTC I/O Registers**

| I/O Locations | If U128E bit = 0          | Function                                       |
|---------------|---------------------------|--|
| 70h and 74h   | Also alias to 72h and 76h | Real-Time Clock (Standard RAM) Index Register  |
| 71h and 75h   | Also alias to 73h and 77h | Real-Time Clock (Standard RAM) Target Register |
| 72h and 76h   |                           | Extended RAM Index Register (if enabled)       |
| 73h and 77h   |                           | Extended RAM Target Register (if enabled)      |

**NOTES:**

1. I/O locations 70h and 71h are the standard ISA location for the real-time clock. The map for this bank is shown in Table 8-7. Locations 72h and 73h are for accessing the extended RAM. The extended RAM bank is also accessed using an indexed scheme. I/O address 72h is used as the address pointer and I/O address 73h is used as the data register. Index addresses above 127h are not valid. If the extended RAM is not needed, it may be disabled.
2. Software must preserve the value of bit 7 at I/O addresses 70h and 74h. When writing to these addresses, software must first read the value, and then write the same value for bit 7 during the sequential address write.

## 8.6.2 Indexed Registers

The RTC contains two sets of indexed registers that are accessed using the two separate Index and Target registers (70/71h or 72/73h), as shown in Table 8-7.

**Table 8-7. RTC (Standard) RAM Bank**

| Index   | Name                  |
|---------|-----------------------|
| 00h     | Seconds               |
| 01h     | Seconds Alarm.        |
| 02h     | Minutes               |
| 03h     | Minutes Alarm         |
| 04h     | Hours                 |
| 05h     | Hours Alarm           |
| 06h     | Day of Week           |
| 07h     | Day of Month          |
| 08h     | Month                 |
| 09h     | Year                  |
| 0Ah     | Register A            |
| 0Bh     | Register B            |
| 0Ch     | Register C            |
| 0Dh     | Register D            |
| 0Eh–7Fh | 114 Bytes of User RAM |

**8.6.2.1 RTC\_REGA—Register A**

|                |           |             |        |
|----------------|-----------|-------------|--------|
| RTC Index:     | 0A        | Attribute:  | R/W    |
| Default Value: | Undefined | Size:       | 8 bits |
| Lockable:      | No        | Power Well: | RTC    |

This register is used for general configuration of the RTC functions. None of the bits are affected by RSMRST# or any other ICH reset signal.

| Bit | Description  |
|-----|--|
| 7   | <b>Update In Progress (UIP).</b> This bit may be monitored as a status flag.<br>1 = The update is soon to occur or is in progress.<br>0 = The update cycle will not start for at least 244us. The time, calendar, and alarm information in RAM is always available when the UIP bit is 0.  |
| 6:4 | <b>Division Chain Select (DV[2:0]).</b> These three bits control the divider chain for the oscillator, and are not affected by RSMRST# or any other reset signal. DV[2] corresponds to bit 6.<br>010 = Normal Operation<br>11X = Divider Reset<br>101 = Bypass 15 stages (test mode only)<br>100 = Bypass 10 stages (test mode only)<br>011 = Bypass 5 stages (test mode only)<br>001 = Invalid<br>000 = Invalid   |
| 3:0 | <b>Rate Select (RS[3:0]).</b> Selects one of 13 taps of the 15 stage divider chain. The selected tap can generate a periodic interrupt if the PIE bit is set in Register B. Otherwise this tap will set the PF flag of Register C. If the periodic interrupt is not to be used, these bits should all be set to zero. RS3 corresponds to bit 3.<br>0000 = Interrupt never toggles<br>0001 = 3.90625 ms<br>0010 = 7.8125 ms<br>0011 = 122.070 us<br>0100 = 244.141 us<br>0101 = 488.281 us<br>0110 = 976.5625 us<br>0111 = 1.953125 ms<br>1000 = 3.90625 ms<br>1001 = 7.8125 ms<br>1010 = 15.625 ms<br>1011 = 31.25 ms<br>1100 = 62.5 ms<br>1101 = 125 ms<br>1110 = 250 ms<br>1111 = 500 ms |



### 8.6.2.2 RTC\_REGB—Register B (General Configuration)

|                |                         |             |        |
|----------------|-------------------------|-------------|--------|
| RTC Index:     | 0Bh                     | Attribute:  | R/W    |
| Default Value: | U0U00UUU (U: Undefined) | Size:       | 8 bits |
| Lockable:      | No                      | Power Well: | RTC    |

| Bit | Description   |
|-----|---|
| 7   | <p><b>Update Cycle Inhibit (SET).</b> Enables/Inhibits the update cycles. This bit is not affected by RSMRST# nor any other reset signal.</p> <p>0 = Update cycle occurs normally once each second.</p> <p>1 = A current update cycle will abort and subsequent update cycles will not occur until SET is returned to zero. When set is one, the BIOS may initialize time and calendar bytes safely.</p>  |
| 6   | <p><b>Periodic Interrupt Enable (PIE).</b></p> <p>1 = The Periodic Interrupt Enable (PIE) bit allows an interrupt to occur with a time base set with the RS bits of register A. This bit is cleared by RSMRST#, but not on any other reset</p>  |
| 5   | <p><b>Alarm Interrupt Enable (AIE).</b></p> <p>1 = The Alarm Interrupt Enable (AIE) bit allows an interrupt to occur when the AF is one as set from an alarm match from the update cycle. An alarm can occur once a second, one an hour, once a day, or one a month. This bit is cleared by RTCRST#, but not on any other reset</p>   |
| 4   | <p><b>Update-ended Interrupt Enable (UIE).</b></p> <p>1 = The Update-ended Interrupt Enable (UIE) bit allows an interrupt to occur when the update cycle ends. This bit is cleared by RSMRST#, but not on any other reset</p>   |
| 3   | <p><b>Square Wave Enable (SQWE).</b> The Square Wave Enable bit serves no function in this device, yet is left in this register bank to provide compatibility with the Motorola* 146818B. There is not SQW pin on this device. This bit is cleared by RSMRST#, but not on any other reset.</p>  |
| 2   | <p><b>Data Mode (DM).</b> The Data Mode (DM) bit specifies either binary or BCD data representation. This bit is not affected by RSMRST# nor any other reset signal.</p> <p>1 = Binary</p> <p>0 = BCD.</p>  |
| 1   | <p><b>Hour Format (HOURFORM).</b> This bit indicates the hour byte format. This bit is not affected by RSMRST# nor any other reset signal.</p> <p>1 = twenty-four hour mode.</p> <p>0 = twelve-hour mode. In twelve hour mode, the seventh bit represents AM as zero and PM as one.</p>   |
| 0   | <p><b>Daylight Savings Enable (DSE).</b> The Daylight Savings Enable bit triggers two special hour updates per year when set to one.</p> <p>1 = a) Update on the first Sunday in April, where time increments from 1:59:59 AM to 3:00:00 AM.<br/>                     b) Update on the last Sunday in October when the time first reaches 1:59:59 AM, it is changed to 1:00:00 AM. The time must increment normally for at least two update cycles (seconds) previous to these conditions for the time change to occur properly.</p> <p>0 = These special update conditions do not occur when the DSE bit is set to zero. The days for the hour adjustment are those specified in United States federal law as of 1987, which is different than previous years. This bit is not affected by RSMRST# nor any other reset signal.</p> |

### 8.6.2.3 RTC\_REGC—Register C (Flag Register)

|                |                         |             |        |
|----------------|-------------------------|-------------|--------|
| RTC Index:     | 0Ch                     | Attribute:  | RO     |
| Default Value: | 00U00000 (U: Undefined) | Size:       | 8 bits |
| Lockable:      | No                      | Power Well: | RTC    |

Writes to Register C have no effect.

| Bit | Description  |
|-----|--|
| 7   | <b>Interrupt Request Flag (IRQF).</b> Interrupt Request Flag = PF * PIE + AF * AIE + UF * UFE. This also causes the CH_IRQ_B signal to be asserted. This bit is cleared upon RSMRST# or a read of Register C.  |
| 6   | <b>Periodic Interrupt Flag (PF).</b> This bit is cleared upon RSMRST# or a read of Register C.<br>1 = Periodic interrupt Flag will be one when the tap as specified by the RS bits of register A is one.<br>0 = If no taps are specified, this flag bit will remain at zero. |
| 5   | <b>Alarm Flag (AF).</b> Alarm Flag will be high after all Alarm values match the current time. This bit is cleared upon RTCRST# or a read of Register C.   |
| 4   | <b>Update-ended Flag (UF).</b> Updated-ended flag will be high immediately following an update cycle for each second. The bit is cleared upon RSMRST# or a read of Register C.   |
| 3:0 | Reserved. Will always report 0.  |

### 8.6.2.4 RTC\_REGD—Register D (Flag Register)

|                |                         |             |        |
|----------------|-------------------------|-------------|--------|
| RTC Index:     | 0Dh                     | Attribute:  | R/W    |
| Default Value: | 10UUUUUU (U: Undefined) | Size:       | 8 bits |
| Lockable:      | No                      | Power Well: | RTC    |

| Bit | Description  |
|-----|--|
| 7   | <b>Valid RAM and Time Bit (VRT).</b><br>1 = The Valid Ram and Time bit is set to one when the PWRGD (power good) signal provided is high. This feature is not typically used.<br>0 = This bit should always be written as a 0 for write cycle, however it will return a 1 for read cycles.   |
| 6   | Reserved. This bit always returns a 0 and should be set to 0 for write cycles.   |
| 5:0 | <b>Date Alarm.</b> These bits store the date of month alarm value. If set to 000000, then a don't care state is assumed. The host must configure the date alarm for these bits to do anything, yet they can be written at any time. If the date alarm is not enabled, these bits will return zeros to mimic the functionality of the Motorola 146818B. These bits are not affected by RESET. |

## 8.7 Processor Interface Registers

### 8.7.1 NMI\_SC—NMI Status and Control Register

|                |     |             |                    |
|----------------|-----|-------------|--------------------|
| I/O Address:   | 61h | Attribute:  | R/W (some bits RO) |
| Default Value: | 00h | Size:       | 8 bits             |
| Lockable:      | No  | Power Well: | Core               |

| Bit | Description   |
|-----|---|
| 7   | <b>SERR# NMI Source Status (SERR#_NMI_STS)—RO.</b><br>1 = PCI agent detected a system error and pulses the PCI SERR# line. This interrupt source is enabled by setting bit 2 to 0. To reset the interrupt, set bit 2 to 1 and then set it to 0. When writing to port 61h, this bit must be 0. |
| 6   | <b>IOCHK# NMI Source Status (IOCHK_NMI_STS) —RO.</b><br>1 = An ISA agent (via SERIRQ) asserted IOCHK# on the ISA bus. This interrupt source is enabled by setting bit 3 to 0. To reset the interrupt, set bit 3 to 0 and then set it to 1. When writing to port 61h, this bit must be a 0.    |
| 5   | <b>Timer Counter 2 OUT Status (TMR2_OUT_STS) —RO.</b> This bit reflects the current state of the 8254 counter 2 output. Counter 2 must be programmed following any PCI reset for this bit to have a determinate value. When writing to port 61h, this bit must be a 0.                        |
| 4   | <b>Refresh Cycle Toggle (REF_TOGGLE) —RO.</b> This signal toggles from either 0 to 1 or 1 to 0 at a rate that is equivalent to when refresh cycles would occur. When writing to port 61h, this bit must be a 0.   |
| 3   | <b>IOCHK# NMI Enable (IOCHK_NMI_EN) —R/W.</b><br>1 = Disabled and cleared.<br>0 = Enabled.  |
| 2   | <b>PCI SERR# Enable (PCI_SERR_EN) —R/W.</b><br>1 = SERR# NMIs are disabled and cleared.<br>0 = SERR# NMIs are enabled.  |
| 1   | <b>Speaker Data Enable (SPKR_DAT_EN) —R/W.</b><br>0 = SPKR output is a 0.<br>1 = SPKR output is equivalent to the Counter 2 OUT signal value.   |
| 0   | <b>Timer Counter 2 Enable (TIM_CNT2_EN) —R/W.</b><br>0 = Disable<br>1 = Enable  |

### 8.7.2 NMI\_EN—NMI Enable (and Real Time Clock Index)

|                |     |             |               |
|----------------|-----|-------------|---------------|
| I/O Address:   | 70h | Attribute:  | R/W (Special) |
| Default Value: | 80h | Size:       | 8 bits        |
| Lockable:      | No  | Power Well: | Core          |

**Note:** The RTC Index field is write-only for normal operation. This field can only be read in Alt-Access Mode. Note, however that this register is aliased to Port 74h (documented in Table 19-2), and all bits are readable at that address.

| Bits | Description  |
|------|--|
| 7    | <b>NMI Enable (NMI_EN).</b><br>1 = Disable All NMI sources.<br>0 = Enable NMI sources.   |
| 6:0  | <b>Real Time Clock Index Address (RTC_INDx).</b> This data goes to the RTC to select which register or CMOS RAM address is being accessed. |

### 8.7.3 PORT92—Fast A20 and Init Register

|                |     |             |        |
|----------------|-----|-------------|--------|
| I/O Address:   | 92h | Attribute:  | R/W    |
| Default Value: | 00h | Size:       | 8 bits |
| Lockable:      | No  | Power Well: | Core   |

| Bit | Description  |
|-----|--|
| 7:2 | Reserved.  |
| 1   | <b>Alternate A20 Gate (ALT_A20_GATE).</b> This bit is Or'd with the A20GATE input signal to generate A20M# to the processor.<br>1 = This bit is set when INIT# goes active.<br>0 = A20M# signal can potentially go active. |
| 0   | <b>INIT_NOW.</b> When this bit transitions from a 0 to a 1, the ICH will force INIT# active for 16 PCI clocks.   |

### 8.7.4 COPROC\_ERR—Coprocessor Error Register

|                |     |             |        |
|----------------|-----|-------------|--------|
| I/O Address:   | F0h | Attribute:  | WO     |
| Default Value: | 00h | Size:       | 8 bits |
| Lockable:      | No  | Power Well: | Core   |

| Bits | Description   |
|------|---|
| 7:0  | <b>COPROC_ERR.</b> Any value written to this register will cause IGNNE# to go active, if FERR# had generated an internal IRQ13. For FERR# to generate an internal IRQ13, the COPROC_ERR_EN bit (Device 31:Function 0, Offset D0, Bit 13) must be 1. |

### 8.7.5 RST\_CNT—Reset Control Register

|                |      |             |        |
|----------------|------|-------------|--------|
| I/O Address:   | CF9h | Attribute:  | R/W    |
| Default Value: | 00h  | Size:       | 8 bits |
| Lockable:      | No   | Power Well: | Core   |

| Bit | Description  |
|-----|--|
| 7:4 | Reserved.  |
| 3   | <p><b>Full Reset (FULL_RST).</b> This bit is used to determine the states of SLP_S3# and SLP_S5# after a CF9 hard reset (SYS_RST = 1 and RST_CPU is set to 1), after PWROK going low (with RSMRST# high), or after two TCO timeouts.</p> <p>1 = ICH will drive SLP_S3# and SLP_S5# low for 3–5 seconds.<br/>                     0 = ICH will keep SLP_S3# and SLP_S5# high.</p> |
| 2   | <p><b>Reset CPU (RST_CPU).</b> When this bit transitions from a '0' to a '1', it initiates a hard or soft reset, as determined by the SYS_RST bit (bit 1 of this register).</p>  |
| 1   | <p><b>System Reset (SYS_RST).</b> This bit is used to determine a hard or soft reset to the processor.</p> <p>1 = When RST_CPU bit goes from 0 to 1, the ICH performs a hard reset by activating PCIRST# for 1 millisecond.<br/>                     0 = When RST_CPU bit goes from 0 to 1, the ICH performs a soft reset by activating INIT# for 16 PCI clocks.</p>             |
| 0   | Reserved.  |

## 8.8 Power Management Registers (D31:F0)

The power management registers are distributed within the PCI Device 31: Function 0 space, as well as a separate I/O range. Each register is described below. Unless otherwise indicate, bits are in the main (core) power well.

Bits not explicitly defined in each register are assumed to be reserved. When writing to a reserved bit, the value should always be 0. Software should not attempt to use the value read from a reserved bit, as it may not be consistently 1 or 0.

### 8.8.1 Power Management PCI Configuration Registers (D31:F0)

Table 8-8 shows a small part of the configuration space for PCI Device 31: Function 0. It includes only those registers dedicated for power management. Some of the registers are only used for Legacy Power management schemes.

Table 8-8. PCI Configuration Map (PM—D31:F0)

| Offset  | Mnemonic    | Register Name/Function                   | Default   | Type |
|---------|-------------|--|-----------|------|
| 40h–43h | PMBASE      | ACPI Base Address (PMBASE)               | 00000000h | R/W  |
| 44h     | ACPI_CNTL   | ACPI Control                             | 00h       | R/W  |
| A0h     | GEN_PMCON_1 | General Power Management Configuration 1 | 0000h     | R/W  |
| A2h     | GEN_PMCON_2 | General Power Management Configuration 2 | 0000h     | R/W  |
| A4h     | GEN_PMCON_3 | General Power Management Configuration 3 | 00h       | R/W  |
| B8–BBh  | GPI_ROUT    | GPI_ROUT                                 | 00000000h | R/W  |
| C4h     | IO_MON_RNG1 | I/O Monitor Range 1                      | 0000h     | R/W  |
| C6h     | IO_MON_RNG2 | I/O Monitor Range 2                      | 0000h     | R/W  |
| CCh     | IO_MON_MSK  | I/O Monitor Range Mask                   | 0000h     | R/W  |

#### 8.8.1.1 GEN\_PMCON\_1—General PM Configuration 1 Register (PM—D31:F0)

|                |     |             |              |
|----------------|-----|-------------|--------------|
| Offset:        | A0h | Attribute:  | R/W          |
| Default Value: | 00h | Size:       | 16 bits      |
| Lockable:      | No  | Usage:      | ACPI, Legacy |
|                |     | Power Well: | Core         |

| Bit   | Description  |
|-------|--|
| 15:10 | Reserved.  |
| 9     | <b>PWRBTN_LVL — RO:</b> This read-only bit indicates the current state of the PWRBTN# signal.<br>1 = High<br>0 = Low.  |
| 8:6   | Reserved.  |
| 5     | <b>CPUSLP_EN:</b> CPU SLP# Enable. Enables the CPUSLP# signal to go active in the S1 state. This reduces the processor power.<br>ICH (82801AA): For dual processor designs, there will be significant power reduction. |
| 4:0   | Reserved.  |

### 8.8.1.2 GEN\_PMCON\_2—General PM Configuration 2 Register (PM—D31:F0)

|                |     |             |              |
|----------------|-----|-------------|--------------|
| Offset:        | A2h | Attribute:  | R/W          |
| Default Value: | 00h | Size:       | 16 bits      |
| Lockable:      | No  | Usage:      | ACPI, Legacy |
|                |     | Power Well: | Resume       |

| Bit | Description  |
|-----|--|
| 7:1 | Reserved.  |
| 0   | <p><b>PWROK_FLR:</b> This bit will be set any time PWROK goes low, when the system was in S0, or S1 state. The bit will be cleared only by software by writing a 1 to this bit or when the system goes to a G3 state.</p> <p><b>NOTE:</b> Traditional designs have a reset button logically AND'd with the PWROK signal from the power supply and the processor's voltage regulator module. If this is done with the ICH, the PWROK_FLR bit will be set. The ICH treats this internally as if the RSMRST# signal had gone active. However, it is not treated as a full power failure. If PWROK goes inactive and then active (but RSMRST# stays high), then the ICH will reboot (regardless of the state of the AFTERG3 bit). If the RSMRST# signal also goes low before PWROK goes high, then this is a full power failure, and the reboot policy is controlled by the AFTERG3 bit.</p> |

### 8.8.1.3 GEN\_PMCON\_3—General PM Configuration 3 Register (PM—D31:F0)

|                |     |             |              |
|----------------|-----|-------------|--------------|
| Offset:        | A4h | Attribute:  | R/W          |
| Default Value: | 00h | Size:       | 8 bits       |
| Lockable:      | No  | Usage:      | ACPI, Legacy |
|                |     | Power Well: | RTC          |

| Bit | Description   |
|-----|---|
| 7:3 | Reserved.   |
| 2   | <p><b>RTC_PWR_STS:</b></p> <p>1 = Indicates that the RTC battery was removed or failed. This bit is set when RTCRST# signal is low.</p> <p>0 = Software clears this bit by writing a 0 to the bit position.</p> <p><b>NOTE:</b> Clearing CMOS in an ICH-based platform can be done by using a jumper on RTCRST# or GPI, or using SAFEMODE strap. Implementations should not attempt to clear CMOS by using a jumper to pull VccRTC low.</p>   |
| 1   | <p><b>PWR_FLR:</b></p> <p>1 = Indicates that the trickle current (from the main battery or trickle supply) was removed or failed. The RSMRST# signal goes active.</p> <p>0 = Indicates that the trickle current has not failed since the last time the bit was cleared. Write 1 to this bit to clear it. This bit is in the RTC well, and is not cleared by any type of reset except RTCRST#.</p> <p><b>NOTE:</b> Clearing CMOS in an ICH-based platform can be done by using a jumper on RTCRST# or GPI, or using SAFEMODE strap. Implementations should not attempt to clear CMOS by using a jumper to pull VccRTC low.</p> |
| 0   | <p><b>AFTERG3_EN:</b> Determines what state to go to when power is re-applied after a power failure (G3 state). In the S5 state, the only enabled wake event is the Power Button or any enabled wake event that was preserved through the power failure. This bit is in the RTC well and is not cleared by any type of reset except writes to CF9h or RTCRST#.</p> <p>0 = System will return to S0 state (boot) after power is re-applied.</p> <p>1 = System will return to the S5 state (except if it was in S4, in which case it will return to S4).</p>  |

### 8.8.1.4 GPI\_ROUT—GPI Routing Control Register (PM—D31:F0)

|                |         |             |         |
|----------------|---------|-------------|---------|
| Offset:        | B8h–BBh | Attribute:  | R/W     |
| Default Value: | 0000h   | Size:       | 32 bits |
| Lockable:      | No      | Power Well: | Resume  |

| Bit   | Description   |
|-------|---|
| 31:30 | GPI[15] Route: Reserved. (Not Implemented)  |
| 29:28 | GPI[14] Route: Reserved. (Not Implemented)  |
| 27:26 | <b>GPI[13] Route:</b> See bits 1:0 for description.   |
| 25:24 | <b>GPI[12] Route:</b> See bits 1:0 for description.   |
| 23:22 | <b>GPI[11] Route:</b> See bits 1:0 for description.   |
| 21:20 | GPI[10] Route: Reserved. (Not Implemented)  |
| 19:18 | <b>GPI[9] Route:</b> See bits 1:0 for description.  |
| 17:16 | <b>GPI[8] Route:</b> See bits 1:0 for description.  |
| 15:14 | <b>GPI[7] Route:</b> See bits 1:0 for description.  |
| 13:12 | <b>GPI[6] Route:</b> See bits 1:0 for description.  |
| 11:10 | <b>GPI[5] Route:</b> See bits 1:0 for description.  |
| 9:8   | GPI[4] Route: Reserved. (Not Implemented)   |
| 7:6   | GPI[3] Route: Reserved. (Not Implemented)   |
| 5:4   | GPI[2] Route: Reserved. (Not Implemented)   |
| 3:2   | <b>GPI[1] Route:</b> See bits 1:0 for description.  |
| 1:0   | <p><b>GPI[0] Route:</b> GPIO[15:0] can be routed to cause an SMI or SCI when the GPI[n]_STS bit is set. If the GPIO is not set to an input, this field has no effect.</p> <p>If the system is in an S1-S5 state and if the GPE1_EN bit is also set, the GPI can cause a Wake event, even if the GPI is NOT routed to cause an SMI# or SCI.</p> <p>00 = No effect.<br/>           01 = SMI# (if corresponding GPE1_EN bit is also set)<br/>           10 = SCI (if corresponding GPE1_EN bit is also set)<br/>           11 = Reserved</p> |

**Note:** GPIOs that are not implemented will not have the corresponding bits implemented in this register.

### 8.8.1.5 IO\_MON\_RNG1—IO Monitor Range 1 Register (PM—D31:F0)

|                |      |            |             |
|----------------|------|------------|-------------|
| Offset:        | C4h  | Attribute: | R/W         |
| Default Value: | 00h  | Size:      | 16 bits     |
| Lockable:      | No   | Usage:     | Legacy Only |
| Power Well:    | Core |            |             |

| Bit  | Description   |
|------|---|
| 15:0 | <b>Used to set the base of I/O range 1 for SMI#.</b> The range can be mapped anywhere in the processor I/O space (0–64 KB). Any access to the range will generate an SMI# if enabled by IOMON1_EN bit in the IOMON_STS_EN register (PMBASE +40h). Note that access to this range will not cause the ICH to trap the cycle |



### 8.8.1.6 IO\_MON\_RNG2—IO Monitor Range 2 Register (PM—D31:F0)

|                |      |            |             |
|----------------|------|------------|-------------|
| Offset:        | C6h  | Attribute: | R/W         |
| Default Value: | 00h  | Size:      | 16 bits     |
| Lockable:      | No   | Usage:     | Legacy Only |
| Power Well:    | Core |            |             |

| Bit  | Description   |
|------|---|
| 15:0 | <b>Used to set the base of I/O range 2 for SMI#.</b> The range can be mapped anywhere in the processor I/O space (0–64 KB). Any access to the range will generate an SMI# if enabled by IOMON2_EN bit in the IOMON_STS_EN register (PMBASE +40h). Note that access to this range will not cause the ICH to trap the cycle |

### 8.8.1.7 IO\_MON\_MSK—IO Monitor Range Mask Register (PM—D31:F0)

|                |      |            |             |
|----------------|------|------------|-------------|
| Offset:        | CCh  | Attribute: | R/W         |
| Default Value: | 00h  | Size:      | 16 bits     |
| Lockable:      | No   | Usage:     | Legacy Only |
| Power Well:    | Core |            |             |

| Bit  | Description  |
|------|--|
| 15:8 | Reserved.  |
| 7:4  | <b>IOMON_RNG2_MSK:</b> Selects the low 4-bit mask for the I/O Monitor Range 2. When a mask bit is set to a 1, the corresponding bit in the base I/O selection will not be decoded. (See bit 3:0 for example).  |
| 3:0  | <b>IOMON_RNG1_MSK:</b> Selects the low 4-bit mask for the I/O Monitor Range 1. When a mask bit is set to a 1, the corresponding bit in the base I/O selection will not be decoded.<br>For example, if IO_MON_RNG1 = 1230h, and IOMON_RNG1_MSK = 0011b, the ICH will decode 1230h, 1231h, 1232h, and 1233h for I/O Monitor Range 1. |

## 8.8.2 APM I/O Decode

Table 8-9 shows the I/O registers associated with APM support. This register space is enabled in the PCI Device 31: Function 0 space (APMDEC\_EN), and cannot be moved (fixed I/O location).

**Table 8-9. APM Register Map**

| Address | Register Name/Function                 | Default | Type |
|---------|--|---------|------|
| B2h     | Advanced Power Management Control Port | 00h     | R/W  |
| B3h     | Advanced Power Management Status Port  | 00h     | R/W  |

### 8.8.2.1 APM\_CNT—Advanced Power Management Control Port Register

|                |      |            |             |
|----------------|------|------------|-------------|
| I/O Address:   | B2h  | Attribute: | R/W         |
| Default Value: | 00h  | Size:      | 8 bits      |
| Lockable:      | No   | Usage:     | Legacy Only |
| Power Well:    | Core |            |             |

| Bit | Description  |
|-----|--|
| 7:0 | <b>Used to pass an APM command between the OS and the SMI handler.</b> Writes to this port not only store data in the APMC register, but also generates an SMI# when the APMC_EN bit is set. |

### 8.8.2.2 APM\_STS—Advanced Power Management Status Port Register

|                |      |            |             |
|----------------|------|------------|-------------|
| I/O Address:   | B3h  | Attribute: | R/W         |
| Default Value: | 00h  | Size:      | 8 bits      |
| Lockable:      | No   | Usage:     | Legacy Only |
| Power Well:    | Core |            |             |

| Bit | Description   |
|-----|---|
| 7:0 | <b>Used to pass data between the OS and the SMI handler.</b> Basically, this is a scratchpad register and is not effected by any other register or function (other than a PCI reset). |

### 8.8.3 Power Management I/O Registers

Table 8-10 shows the registers associated with ACPI and Legacy power management support. These registers are enabled in the PCI Device 31: Function 0 space (PM\_IO\_EN), and can be moved to any I/O location (128-byte aligned). The registers are defined to be compliant with the ACPI 1.0 specification, and use the same bit names.

**Note:** All reserved bits and registers will always return 0 when read, and will have no effect when written.

**Table 8-10. ACPI and Legacy I/O Register Map**

| PMBASE+ Offset | Register Name/Function                                    | Default    | Attributes |
|----------------|---|------------|------------|
| 00–01h         | PM1 Status, ACPI Pointer: PM1a_EVT_BLK                    | 0000h      | R/W        |
| 02–03h         | PM1 Enable, ACPI Pointer: PM1a_EVT_BLK+2                  | 0000h      | R/W        |
| 04–07h         | PM1 Control, ACPI Pointer: PM1a_CNT_BLK                   | 00000000h  | R/W        |
| 08–0Bh         | PM1 Timer, ACPI Pointer: PMTMR_BLK                        | 00000000h  | RO         |
| 0Ch–0Fh        | Reserved  |            |            |
| 10h–13h        | Processor Control, ACPI Pointer: P_BLK                    | 00000000h  | R/W        |
| 14h            | Level 2 Register, ACPI Pointer: P_BLK+4                   | 00h        | RO         |
| 16–20h         | Reserved  |            |            |
| 28–29h         | General Purpose Event 0 Status, ACPI Pointer: GPE0_BLK    | 0000h      | R/W        |
| 2A–2Bh         | General Purpose Event 0 Enables, ACPI Pointer: GPE0_BLK+2 | 0000h      | R/W        |
| 2C–2D          | General Purpose Event 1 Status, ACPI Pointer: GPE1_BLK    | 0000h      | R/W        |
| 2E–2F          | General Purpose Event 1 Enables, ACPI Pointer: GPE1_BLK+2 | 0000h      | R/W        |
| 30–31h         | SMI# Control and Enable                                   | 0000h      | R/W        |
| 34–35h         | SMI Status Register                                       | 0000h      | R/W        |
| 40h            | I/O Monitor Status and Enable                             | 0000h      | R/W        |
| 44h            | Device Activity Status                                    | 0000h      | R/W        |
| 4Ch–4Dh        | Bus Address Tracker                                       | Last Cycle | Read-Only  |
| 4Eh            | Bus Cycle Tracker   | Last Cycle | Read-Only  |
| 50–5F          | Reserved.   | 00h        | R/W        |
| 61–7Bh         | Reserved for Legacy Power Management.                     | 00h        | Read-Only  |
| 7Ch–7Fh        | Reserved  | 00h        | Read-Only  |

### 8.8.3.1 PM1\_STS—Power Management 1 Status Register

|                |  |            |                |
|----------------|--|------------|----------------|
| I/O Address:   | PMBASE + 00h<br>(ACPI PM1a_EVT_BLK)                          | Attribute: | R/W            |
| Default Value: | 0000h  | Size:      | 16 bits        |
| Lockable:      | No   | Usage:     | ACPI or Legacy |
| Power Well:    | Bits 0:7: Core,<br>Bits 8:15: Resume,<br>Bit 11 still in RTC |            |                |

If bit 10 or 8 in this register is set, and the corresponding \_EN bit is set in the PM1\_EN register, then the ICH will generate a Wake Event. Once back in an S0 state (or if already in an S0 state when the event occurs), the ICH will also generate an SCI if the SCI\_EN bit is set, or an SMI# if the SCI\_EN bit is not set.

**Note:** Bit 5 does not cause an SMI# or a wake event. Bit 0 does not cause a wake event but can cause an SMI# or SCI.

| Bit   | Description   |
|-------|---|
| 15    | <p><b>WAK_STS.</b></p> <p>1 = This bit is set when the system is in one of the sleep states (via the SLP_EN bit) and an enabled wake event occurs. Upon setting this bit, the ICH will transition the system to the ON state. This bit is only set by hardware.</p> <p>0 = Cleared by writing a one to this bit. This bit is not affected by hard resets caused by a CF9 write, but is reset by RSMRST#.</p> <p><b>NOTE:</b> If a power failure occurs (such as removed batteries) without the SLP_EN bit set, the WAK_STS bit will not be set when power returns.</p>  |
| 14:12 | Reserved  |
| 11    | <p><b>PRBTNOR_STS.</b></p> <p>1 = This bit is set by hardware anytime a Power Button Override Event occurs, which occurs when the power button is pressed for at least 4 consecutive seconds. The power button override causes an unconditional transition to the S5 state, as well as set the AFTERG3 bit.</p> <p>0 = The BIOS or SCI handler can clear this bit by writing a 1 to it.</p> <p><b>NOTE:</b> This bit is not affected by hard resets caused by a CF9 write, and will be preserved through a power failure.</p>   |
| 10    | <p><b>RTC_STS.</b></p> <p>1 = This bit is set by hardware when the RTC generates an alarm (assertion of the IRQ8# signal). Additionally, if the RTC_EN bit is set, the setting of the RTC_STS bit will generate a wake event.</p> <p>0 = Cleared by writing a 1 to this bit position.</p> <p><b>NOTE:</b> This bit is not affected by hard resets caused by a CF9 write, but is reset by RSMRST#.</p>   |
| 9     | Reserved  |
| 8     | <p><b>PWRBTN_STS.</b> This bit is set by hardware when the PWRBTN# signal is asserted Low, independent of any other enable bit. This bit is not affected by hard resets caused by a CF9 write. In the S0 state, while PWRBTN_EN and PWRBTN_STS are both set, an SCI (or SMI# if SCI_EN is not set) will be generated.</p> <p>In any sleeping state S1-S5, while PWRBTN_EN and PWRBTN_STS are both set, a wake event is generated.</p> <p>If the PWRBTN# signal is held low for more than 4 seconds, the hardware clears the PWRBTN_STS bit, sets the PWRBTNOR_STS bit, the system transitions to the S5 state, and only PWRBTN# is enabled as a wake event.</p> <p>This bit can be cleared by software by writing a one to this bit position.</p> |
| 7:6   | Reserved  |

| Bit | Description   |
|-----|---|
| 5   | <b>GBL_STS.</b><br>1 = This bit is set when an SCI is generated due to BIOS wanting the attention of the SCI handler. BIOS has a corresponding bit, BIOS_RLS, which will cause an SCI and set this bit.<br>0 = The SCI handler should then clear this bit by writing a 1 to it.   |
| 4:1 | Reserved  |
| 0   | <b>TMROF_STS.</b> This is the timer overflow status bit.<br>1 = This bit gets set anytime the 22 <sup>nd</sup> bit of the 24 bit timer goes high (bits are counted from 0 to 23). This will occur every 2.3435 seconds. When the TMROF_EN bit is set, then the setting of the TMROF_STS bit will additionally generate an SCI or SMI# (depending on the SCI_EN).<br>0 = The SCI or SMI# handler clears this bit by writing a 1 to this bit. |

### 8.8.3.2 PM1\_EN—Power Management 1 Enables Register

|                |  |            |                |
|----------------|--|------------|----------------|
| I/O Address:   | PMBASE + 02h<br>( <i>ACPI PM1a_EVT_BLK + 2</i> ) | Attribute: | R/W            |
| Default Value: | 0000h  | Size:      | 16 bits        |
| Lockable:      | No   | Usage:     | ACPI or Legacy |
| Power Well:    | Bits 0:7: Core,<br>Bits 8:15: Resume             |            |                |

| Bit      | Description  |                              |        |                              |   |   |                |   |   |      |   |   |     |
|----------|--|------------------------------|--------|------------------------------|---|---|----------------|---|---|------|---|---|-----|
| 15:11    | Reserved.  |                              |        |                              |   |   |                |   |   |      |   |   |     |
| 10       | <b>RTC_EN.</b><br>1 = Enable. An SCI (or SMI#) or wake event will occur when this bit is set and the RTC_STS bit goes active. If this bit is not set, then setting the RTC_STS bit does not cause an SCI (or SMI#) or wake event.<br>0 = This bit is not cleared by any reset other than RTCRST# and a power button override event.<br><b>NOTE:</b> This bit is in the RTC well to allow an RTC event to wake after a power failure.             |                              |        |                              |   |   |                |   |   |      |   |   |     |
| 9        | Reserved   |                              |        |                              |   |   |                |   |   |      |   |   |     |
| 8        | <b>PWRBTN_EN.</b> This bit is used to enable the setting of the PWRBTN_STS bit to generate a power management event (SMI#, SCI). PWRBTN_EN has no effect on the PWRBTN_STS bit being set by the assertion of the power button. The Power Button is always enabled as a Wake event.<br>1 = Enable<br>0 = Disable  |                              |        |                              |   |   |                |   |   |      |   |   |     |
| 7:6      | Reserved.  |                              |        |                              |   |   |                |   |   |      |   |   |     |
| 5        | <b>GBL_EN.</b> The Global Enable bit. When both the GBL_EN and the GBL_STS are set, an SCI is raised.<br>1 = Enable<br>0 = Disable   |                              |        |                              |   |   |                |   |   |      |   |   |     |
| 4:1      | Reserved.  |                              |        |                              |   |   |                |   |   |      |   |   |     |
| 0        | <b>TMROF_EN.</b> This is the timer overflow interrupt enable bit and works in conjunction with the SCI_EN bit: <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>TMROF_EN</th> <th>SCI_EN</th> <th>Effect when TMROF_STS is set</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>x</td> <td>No SMI# or SCI</td> </tr> <tr> <td>1</td> <td>0</td> <td>SMI#</td> </tr> <tr> <td>1</td> <td>1</td> <td>SCI</td> </tr> </tbody> </table> | TMROF_EN                     | SCI_EN | Effect when TMROF_STS is set | 0 | x | No SMI# or SCI | 1 | 0 | SMI# | 1 | 1 | SCI |
| TMROF_EN | SCI_EN   | Effect when TMROF_STS is set |        |                              |   |   |                |   |   |      |   |   |     |
| 0        | x  | No SMI# or SCI               |        |                              |   |   |                |   |   |      |   |   |     |
| 1        | 0  | SMI#                         |        |                              |   |   |                |   |   |      |   |   |     |
| 1        | 1  | SCI                          |        |                              |   |   |                |   |   |      |   |   |     |

### 8.8.3.3 PM1\_CNT—Power Management 1 Control

I/O Address: PMBASE + 04h  
 (ACPI PM1a\_CNT\_BLK) Attribute: R/W  
 Default Value: 0000h Size: 32 bits  
 Lockable: No Usage: ACPI or Legacy  
 Power Well: Bits 0:7: Core,  
 Bits 8:15: Resume

| Bit   | Description   |
|-------|---|
| 13    | <b>SLP_EN — WO.</b> This is a write-only bit and reads to it always return a zero.<br>1 = Setting this bit causes the system to sequence into the Sleep state defined by the SLP_TYP field.   |
| 12:10 | <b>SLP_TYP.</b> This 3-bit field defines the type of Sleep the system should enter when the SLP_EN bit is set to 1.<br>000 = ON<br>001 = Just assert STPCLK#. Puts processor in Stop-Grant state. Also assert CPUSLP#, to put processor in sleep state.<br>010 = Reserved<br>011 = Reserved<br>100 = Reserved<br>101 = Suspend-To-RAM. Assert SLP_S3#.<br>110 = Suspend-To-Disk. Assert SLP_S3# and , SLP_S5#.<br>111 = Soft Off. Assert SLP_S3#, and SLP_S5# |
| 2     | <b>GBL_RLS.</b> This bit is used by the ACPI software to raise an event to the BIOS. BIOS software has a corresponding enable and status bits to control its ability to receive ACPI events. This bit always reads as 0.  |
| 1     | Reserved  |
| 0     | <b>SCI_EN.</b> Selects the SCI interrupt or the SMI interrupt for various events including the bits in the PM1_STS register (bit 10, 8, 0), and bits in GPE0_STS.<br>1 = These events will generate an SCI interrupt.<br>0 = These events will generate an SMI#.  |

### 8.8.3.4 PM1\_TMR—Power Management 1 Timer Register

I/O Address: PMBASE + 08h  
 (ACPI PMTMR\_BLK) Attribute: RO  
 Default Value: xx000000h Size: 32 bits  
 Lockable: No Usage: ACPI  
 Power Well: Core

| Bit   | Description   |
|-------|---|
| 31:24 | Reserved  |
| 23:0  | <b>TMR_VAL.</b> This read-only field returns the running count of the PM timer. This counter runs off a 3.579545 MHz clock (14.31818 MHz divided by 4). It is reset to zero during a PCI reset, and then continues counting as long as the system is in the S0 state.<br>Anytime the 22 <sup>nd</sup> bit of the timer goes HIGH to LOW (bits referenced from 0 to 23), the TMROF_STS bit is set. The High-to-Low transition will occur every 2.3435 seconds. If the TMROF_EN bit is set, an SCI interrupt is also generated. Writes to this register have no effect. |

### 8.8.3.5 PROC\_CNT—Processor Control Register

|                |                                       |            |                |
|----------------|---------------------------------------|------------|----------------|
| I/O Address:   | PMBASE + 10h<br>( <b>ACPI P_BLK</b> ) | Attribute: | R/W            |
| Default Value: | 00000000h                             | Size:      | 32 bits        |
| Lockable:      | No (bits 8:6 are write once)          | Usage:     | ACPI or Legacy |
| Power Well:    | Core                                  |            |                |

| Bit      | Description   |            |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
|----------|---|------------|---------------|------------|-----|-------------------------------------|-----|-----|-------|-----|-----|-------|-----|-----|-------|-----|-----|-----|-----|-----|-------|-----|-----|-----|-----|-----|-------|-----|
| 31:18    | Reserved.   |            |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 17       | <p><b>THTL_STS—RO.</b> This bit indicates if the clock state machine is in a low power state.<br/>1 = Indicates that the clock state machine is in some type of low power state (where the processor is not running at its maximum performance): thermal throttling or hardware throttling.</p>   |            |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 7:5      | <p><b>THRM_DTY.</b> This write-once 3-bit field determines the duty cycle of the throttling when the thermal override condition occurs. The duty cycle indicates the approximate percentage of time the STPCLK# signal is asserted while in the throttle mode. The STPCLK# throttle period is 1024 PCICLKs. Note that the throttling only occurs if the system is in the C0 state. If in the C2 state, no throttling occurs.</p> <p>There is no enable bit for thermal throttling, because it should not be disabled. Once the THRM_DTY field is written, any subsequent writes will have no effect until PCIRST# goes active.</p> <table border="1"> <thead> <tr> <th>THRM_DTY</th> <th>Throttle Mode</th> <th>PCI Clocks</th> </tr> </thead> <tbody> <tr> <td>000</td> <td>RESERVED (Default)<br/>(Will be 50%)</td> <td>512</td> </tr> <tr> <td>001</td> <td>87.5%</td> <td>896</td> </tr> <tr> <td>010</td> <td>75.0%</td> <td>768</td> </tr> <tr> <td>011</td> <td>62.5%</td> <td>640</td> </tr> <tr> <td>100</td> <td>50%</td> <td>512</td> </tr> <tr> <td>101</td> <td>37.5%</td> <td>384</td> </tr> <tr> <td>110</td> <td>25%</td> <td>256</td> </tr> <tr> <td>111</td> <td>12.5%</td> <td>128</td> </tr> </tbody> </table> | THRM_DTY   | Throttle Mode | PCI Clocks | 000 | RESERVED (Default)<br>(Will be 50%) | 512 | 001 | 87.5% | 896 | 010 | 75.0% | 768 | 011 | 62.5% | 640 | 100 | 50% | 512 | 101 | 37.5% | 384 | 110 | 25% | 256 | 111 | 12.5% | 128 |
| THRM_DTY | Throttle Mode   | PCI Clocks |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 000      | RESERVED (Default)<br>(Will be 50%)   | 512        |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 001      | 87.5%   | 896        |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 010      | 75.0%   | 768        |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 011      | 62.5%   | 640        |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 100      | 50%   | 512        |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 101      | 37.5%   | 384        |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 110      | 25%   | 256        |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 111      | 12.5%   | 128        |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 4        | <p><b>THTT_EN.</b> When set and the system is in a C0 state, it enables a processor-controlled STPCLK# throttling. The duty cycle is selected in the THTL_DTY field.<br/>1 = Enable<br/>0 = Disable</p>   |            |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 3:1      | <p><b>THTL_DTY.</b> This 3-bit field determines the duty cycle of the throttling when the THTL_EN bit is set. The duty cycle indicates the approximate percentage of time the STPCLK# signal is asserted (low) while in the throttle mode. The STPCLK# throttle period is 1024 PCICLKs.</p> <table border="1"> <thead> <tr> <th>THTL_DTY</th> <th>Throttle Mode</th> <th>PCI Clocks</th> </tr> </thead> <tbody> <tr> <td>000</td> <td>RESERVED (Default)<br/>(Will be 50%)</td> <td>512</td> </tr> <tr> <td>001</td> <td>87.5%</td> <td>896</td> </tr> <tr> <td>010</td> <td>75.0%</td> <td>768</td> </tr> <tr> <td>011</td> <td>62.5%</td> <td>640</td> </tr> <tr> <td>100</td> <td>50%</td> <td>512</td> </tr> <tr> <td>101</td> <td>37.5%</td> <td>384</td> </tr> <tr> <td>110</td> <td>25%</td> <td>256</td> </tr> <tr> <td>111</td> <td>12.5%</td> <td>128</td> </tr> </tbody> </table>  | THTL_DTY   | Throttle Mode | PCI Clocks | 000 | RESERVED (Default)<br>(Will be 50%) | 512 | 001 | 87.5% | 896 | 010 | 75.0% | 768 | 011 | 62.5% | 640 | 100 | 50% | 512 | 101 | 37.5% | 384 | 110 | 25% | 256 | 111 | 12.5% | 128 |
| THTL_DTY | Throttle Mode   | PCI Clocks |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 000      | RESERVED (Default)<br>(Will be 50%)   | 512        |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 001      | 87.5%   | 896        |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 010      | 75.0%   | 768        |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 011      | 62.5%   | 640        |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 100      | 50%   | 512        |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 101      | 37.5%   | 384        |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 110      | 25%   | 256        |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 111      | 12.5%   | 128        |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |
| 0        | Reserved  |            |               |            |     |                                     |     |     |       |     |     |       |     |     |       |     |     |     |     |     |       |     |     |     |     |     |       |     |

**8.8.3.6 LV2 — Level 2 Register**

|                |   |            |                |
|----------------|---|------------|----------------|
| I/O Address:   | PMBASE + 14h<br>( <i>ACPI P_BLK+4</i> ) | Attribute: | RO             |
| Default Value: | 00h                                     | Size:      | 8 bits         |
| Lockable:      | No                                      | Usage:     | ACPI or Legacy |
| Power Well:    | Core                                    |            |                |

| Bit | Description  |
|-----|--|
| 7:0 | Reads to this register return all zeros, writes to this register have no effect. Reads to this register generate a "enter a level 2 power state" (C2) to the clock control logic. This will cause the STPCLK# signal to go active, and stay active until a break event occurs. Throttling (due either to THTL_EN or THRM# override) will be ignored. |

**8.8.3.7 GPE0\_STS—General Purpose Event 0 Status Register**

|                |  |            |         |
|----------------|--|------------|---------|
| I/O Address:   | PMBASE + 28h<br>( <i>ACPI GPE0_BLK</i> ) | Attribute: | R/W     |
| Default Value: | 0000h                                    | Size:      | 16 bits |
| Lockable:      | No                                       | Usage:     | ACPI    |
| Power Well:    | Resume                                   |            |         |

**Note:** This register is symmetrical to the General Purpose Event 0 Enable Register. If the corresponding \_EN bit is set, then when the \_STS bit get set, the ICH generates a Wake Event. Once back in an S0 state (or if already in an S0 state when the event occurs), the ICH will also generate an SCI if the SCI\_EN bit is set, or an SMI# if the SCI\_EN bit is not set. No SCI/SMI# or wake event on THRMOR\_STS since no corresponding \_EN bit. None of these bits should be reset by CF9h write. All should be reset by RSMRST#, except for RI\_STS.

| Bit   | Description   |
|-------|---|
| 15:12 | Reserved.   |
| 11    | <b>PME_STS.</b><br>1 = This bit is set to 1 by hardware when the PME# signal goes active. Additionally, if the PME_EN bit is set, and the system is in an S0 state, then the setting of the PME_STS bit will generate an SCI or SMI# (if SCI_EN is not set). If the PME_EN bit is set, and the system is in an S1-S4 state (or S5 state due to setting SLP_TYP and SLP_EN), then the setting of the PME_STS bit will generate a wake event, and an SCI will be generated. If the system is in an S5 state due to power button override or a power failure, then PME_STS will not cause a wake event or SCI. |
| 10:9  | Reserved.   |
| 8     | <b>RI_STS.</b><br>1 = This bit is set to 1 by hardware when the Ri# input signal goes active. The value of this bit must be maintained, even through a G3 state.<br>0 = This bit can be reset by writing a one to this bit position.<br><b>NOTE:</b> This bit is not effected by a hard reset caused by a CF9h write.   |
| 7     | <b>SMB_WAK_STS.</b><br>1 = This bit is set to 1 by hardware to indicate that the wake event was caused by the ICH's SMBus logic.<br>0 = This bit is cleared by writing back a 1 to this bit position.<br>The SMBus controller will independently cause an SMI# or SCI, so this bit does not need to do so (unlike the other bits in this register). This bit is in resume well.   |
| 6     | <b>TCOSCI_STS.</b><br>1 = This bit will be set to 1 by hardware when the TCO logic causes an SCI.<br>0 = This bit can be reset by writing a one to this bit position.   |



| Bit | Description  |
|-----|--|
| 5   | <b>AC97_STS.</b><br>1 = This bit will be set to 1 by hardware when the codecs are attempting to wake the system. The value of this bit must be maintained, even through a G3 state. The AC97_STS bit gets set only from the following two cases:<br>1.ACSDIN[1] or ACSDIN[0] is high and BITCLK is not oscillating, or<br>2.The GSCI bit is set (section 13.2.9, NAMBAR +30h, bit 0)<br>0 = This bit can be reset by writing a one to this bit position.<br>This bit is not effected by a hard reset caused by a CF9h write. |
| 4   | Reserved.  |
| 3   | <b>USB_STS.</b><br>1 = This bit is set when USB controller needs to cause a wake. Additionally if the USB_EN bit is set, the setting of the USB_STS bit will generate a wake event. This bit is set only by hardware<br>0 = Reset by writing a one to this position or a resume-well reset.  |
| 2   | Reserved.  |
| 1   | <b>THRMOR_STS.</b> This is the thermal interrupt over-ride status.<br>1 = This bit is set by hardware anytime a thermal over-ride condition occurs and starts throttling the processor's clock at the THRM_DTY ratio.<br>0 = This bit is cleared by writing a one to this bit position or a resume-well reset. This will not cause an SMI#, SCI, or wake event.  |
| 0   | <b>THRM_STS.</b> This is the thermal interrupt status bit.<br>1 = This bit is set by hardware anytime the THRM# signal is driven active as defined by the THRM_POL bit. Additionally if the THRM_EN bit is set, then the setting of the THRM_STS bit will additionally generate a power management event (SCI or SMI#).<br>0 = This bit is cleared by software by writing a one to this bit position or a resume-well reset.   |

### 8.8.3.8 GPE0\_EN—General Purpose Event 0 Enables Register

|                |                                     |            |         |
|----------------|-------------------------------------|------------|---------|
| I/O Address:   | PMBASE + 2Ah<br>(ACPI GPE0_BLK + 2) | Attribute: | R/W     |
| Default Value: | 0000h                               | Size:      | 16 bits |
| Lockable:      | No                                  | Usage:     | ACPI    |
| Power Well:    | Bits 0:7 Resume,<br>Bits 8:15 RTC   |            |         |

**Note:** This register is symmetrical to the General Purpose Event 0 Status Register. All the bits in this register should be cleared to 0 based on a Power Button Override. The resume well bits are all cleared by RSMRST#.

| Bit   | Description   |
|-------|---|
| 15:12 | Reserved.   |
| 11    | <b>PME_EN.</b> Enables the setting of the PME_STS to generate a wake event and/or an SCI. PME# can be a wake event from the S1–S4 state or from S5 (if entered via SLP_EN, but not power button override).<br>1 = Enable<br>0 = Disable. This bit is only cleared by software, RTCRST#, or RSMRST#. |
| 10:9  | Reserved  |
| 8     | <b>RI_EN.</b> When both RI_EN and RI_STS are set, a Wake event will occur. If RI_EN is not set, then when RI_STS is set, no Wake event will occur. This bit is only cleared by software or RTCRST#.   |

| Bit | Description   |
|-----|---|
| 7   | <b>SMB_WAK_EN.</b> This bit enables the ICH's SMBus controller to generate a wake event. This bit is in the resume well.<br>1 = Enable<br>0 = Disable   |
| 6   | <b>TCOSCI_EN.</b> When TCOSCI_EN and TCOSCI_STS are both set, an SCI will be generated. This bit is in the resume well.<br>1 = Enable<br>0 = Disable  |
| 5   | <b>AC97_EN.</b> When both AC97_EN and AC97_STS bits are set, a Wake event will occur. If AC97_EN is not set, then when AC97_STS is set, no Wake event will occur.<br>0 = Disable  |
| 4   | Reserved  |
| 3   | <b>USB_EN.</b> This bit is used to enable the setting of the USB_STS bit to generate a wake event. The USB_STS bit is set anytime USB signals a wake event. Break events are handled via the interrupt.<br>1 = Enable<br>0 = Disable                        |
| 2   | <b>THRM#_POL.</b> This bit controls the polarity of the THRM# pin needed to set the THRM_STS bit.<br>1 = HIGH value on the THRM# signal will set the THRM_STS bit.<br>0 = Low value on the THRM# signal will set the THRM_STS bit.                          |
| 1   | Reserved.   |
| 0   | <b>THRM_EN.</b> This is the thermal enable bit. When this bit is set an active assertion of the THRM# signal (as defined by the THRM_POL bit) will set the THRM_STS bit and generate a power management event (an SCI or SMI).<br>1 = Enable<br>0 = Disable |

### 8.8.3.9 GPE1\_STS—General Purpose Event 1 Status Register

|                |                                 |            |         |
|----------------|---------------------------------|------------|---------|
| I/O Address:   | PMBASE + 2Ch<br>(ACPI GPE1_BLK) | Attribute: | R/W     |
| Default Value: | 0000h                           | Size:      | 16 bits |
| Lockable:      | No                              | Usage:     | ACPI    |
| Power Well:    | Resume                          |            |         |

This register is symmetrical to the General Purpose Event 1 Enable Register. GPIOs that are not implemented will not have the corresponding bits implemented in this register.

| Bit  | Description   |
|------|---|
| 15:0 | <b>GPI[n]_STS.</b><br>1 = These bits are set any time the corresponding GPIO is set up as an input and the corresponding GPIO signal is low (or high if the corresponding GP_INV bit is set). If the corresponding GPI[n]_EN bit is set in the GPE1_EN register, and the GPI[n]_STS bit is set, then: <ul style="list-style-type: none"> <li>If the system is in an S1_S5 state, the event will also wake the system.</li> <li>If the system is in an S0 state (or upon waking back to an S0 state), an SMI# or SCI will be caused, depending on the GPI_ROUT bits for the corresponding GPI.</li> </ul> 0 = Each bit is cleared by writing a 1 to the bit position when the corresponding GPIO signal is not active. (The status bit will not be cleared if writing a 1 to the bit position while the signal is still active). |

### 8.8.3.10 GPE1\_EN—General Purpose Event 1 Enable Register

|                |  |            |         |
|----------------|--|------------|---------|
| I/O Address:   | PMBASE + 2Eh<br>( <b>ACPI GPE1_BLK + 2</b> ) | Attribute: | R/W     |
| Default Value: | 0000h  | Size:      | 16 bits |
| Lockable:      | No   | Usage:     | ACPI    |
| Power Well:    | Resume                                       |            |         |

**Note:** This register is symmetrical to the General Purpose Event 1 Status Register. GPIOs that are not implemented will not have the corresponding bits implemented in this register.

| Bit  | Description   |
|------|---|
| 15:0 | <b>GPI[n]_EN.</b> These bits enable the corresponding GPI[n]_STS bits being set to cause an SMI#, SCI, and/or wake event. |

### 8.8.3.11 SMI\_EN—SMI Control and Enable Register

|                |              |            |                |
|----------------|--------------|------------|----------------|
| I/O Address:   | PMBASE + 30h | Attribute: | R/W            |
| Default Value: | 0000h        | Size:      | 16 bit         |
| Lockable:      | No           | Usage:     | ACPI or Legacy |
| Power Well:    | Core         |            |                |

| Bit  | Description   |
|------|---|
| 15   | Reserved  |
| 14   | <b>1MIN_EN.</b><br>1 = Enables the ICH to generate an SMI# when the 1MIN_STS bit is set in the SMI_STS register.  |
| 13   | <b>TCO_EN.</b><br>1 = Enables the TCO logic to generate SMI#.<br>0 = Disables TCO logic generating an SMI#.<br><br><b>NOTE:</b> If the NMI2SMI_EN bit is set, SMIs that are caused by re-routed NMIs will not be gated by the TCO_EN bit. Even if the TCO_EN bit is 0, NMIs will still be routed to cause SMIs.   |
| 12:8 | Reserved  |
| 7    | <b>BIOS_RLS.</b><br>1 = Enables the generation of an SCI interrupt for ACPI software when a one is written to this bit position by BIOS software. This bit always reads a zero.   |
| 6    | <b>SWSMI_TMR_EN.</b><br>1 = Starts Software SMI# Timer. When the 64 ms timer expires ( $\pm 4$ ms), it will generate an SMI# and set the SWSMI_TMR_STS bit. The SWSMI_TMR_EN bit will remain at 1 until software sets it back to 0.<br>0 = Disable. Clearing the SWSMI_TMR_EN bit before the timer expires will reset the timer and the SMI# will not be generated. The default for this bit is 0.                                      |
| 5    | <b>APMC_EN.</b><br>1 = Enables writes to the APMC register to cause an SMI#   |
| 4    | <b>SLP_SMI_EN.</b><br>1 = When this bit is set, any write of a 1 to the SLP_EN bit (bit 13 in PM1_CNT register) will generate an SMI#, and the system will not transition to the sleep state based on that write to the SLP_EN bit.<br>0 = Disables the generation of SMI# on SLP_EN.<br><br><b>NOTE:</b> This bit must be 0 before the software attempts to transition the system into a sleep state by writing a 1 to the SLP_EN bit. |

| Bit | Description  |
|-----|--|
| 3   | <b>LEGACY_USB_EN.</b><br>1 = Enables legacy USB circuit to cause SMI#.   |
| 2   | <b>BIOS_EN.</b><br>1 = Enables the generation of SMI# when ACPI software writes a 1 to the GBL_RLS bit.  |
| 1   | <b>EOS: End of SMI.</b> This bit controls the arbitration of the SMI signal to the processor. This bit must be set for the ICH to assert SMI# low to the processor. When this bit is set, SMI# signal will be deasserted for 4 PCI clocks before its assertion. Once the ICH asserts SMI# low, the EOS bit is automatically cleared. In the SMI handler, the processor should clear all pending SMIs (by servicing them and then clearing their respective status bits), set the EOS bit, and exit SMM. This will allow the SMI arbiter to re-assert SMI upon detection of an SMI event and the setting of a SMI status bit. |
| 0   | <b>GBL_SMI_EN.</b><br>1 = Enables the generation of SMIs in the system upon any enabled SMI event.<br>0 = This bit is reset by a PCI reset event. If this bit is not set, no SMI# will be generated.   |

### 8.8.3.12 SMI\_STS—SMI Status Register

|                |              |            |                |
|----------------|--------------|------------|----------------|
| I/O Address:   | PMBASE + 34h | Attribute: | R/W            |
| Default Value: | 0000h        | Size:      | 16 bits        |
| Lockable:      | No           | Usage:     | ACPI or Legacy |
| Power Well:    | Core         |            |                |

**Note:** If the corresponding \_EN bit is set when the \_STS bit is set, the ICH will cause an SMI# (except bits 8-10 and 12, which don't need enable bits since they are logic ORs of other registers that have enable bits).

| Bit | Description   |
|-----|---|
| 15  | <b>SERIRQ_SMI_STS.</b><br>1 = Indicates that the SMI# was caused by the SERIRQ decoder.<br>0 = SMI# was not caused by SERIRQ decoder. This is not a sticky bit.   |
| 14  | <b>1MIN_STS.</b><br>1 = This is set once per minute ( $\pm 4$ seconds). If the 1MIN_EN bit is also set, the ICH will generate an SMI#.<br>0 = This bit is cleared by writing a 1 to its bit position.   |
| 13  | <b>TCO_STS.</b><br>1 = Indicates the SMI# was caused by the TCO logic.<br>0 = SMI# not caused by TCO logic.   |
| 12  | <b>DEVMON_STS.</b> This bit is a logical OR of bit 12 and 13 in IOMON_STS_EN and DEVACT_STS registers and is not a sticky bit.<br>1 = Indicates that the device monitoring logic to generate SMI#. The specific cause of the SMI# is indicated in the IOMON_STS_EN register and DEVACT_STS register.                                  |
| 11  | Reserved  |
| 10  | <b>GPE1_STS.</b> This bit is a logical OR of the bits in the GPE1_STS register that are also set up to cause an SMI# (as indicated by the GPI_ROUT registers) and have the corresponding bit set in the GPE1_EN register. Bits that are not routed to cause an SMI# will have no effect on the GPE1_STS bit. This is not a sticky bit |
| 9   | <b>GPE0_STS.</b> This bit is a logical OR of the bits in the GPE0_STS register that also have the corresponding bit set in the GPE0_EN register. This bit is NOT sticky.  |
| 8   | <b>PM1_STS_REG.</b> This is an OR of the bits (except bit 5) in the ACPI PM1 Status Reg. (offset PMBASE+00h). Not sticky.   |
| 7   | Reserved.   |

| Bit | Description  |
|-----|--|
| 6   | <b>SWSMI_TMR_STS.</b><br>1 = Set by the hardware when the Software SMI# Timer expires.<br>0 = This bit will remain 1 until the software writes a 1 to this bit.  |
| 5   | <b>APM_STS.</b> SMI# was generated by a write access to the APM control register and if the APMC_EN bit is set.<br>0 = Cleared by writing a 1 to its bit position.   |
| 4   | <b>SLP_SMI_STS.</b><br>1 = Indicates an SMI# was caused by a write of 1 to SLP_EN bit when SLP_SMI_EN bit is also set.<br>0 = Cleared by software writing a 1 to the bit position.   |
| 3   | <b>LEGACY_USB_STS.</b> This non-sticky bit is a logical OR of each of the SMI status bits in the USB Legacy key board Register ANDed with the corresponding enable bits. This bit will not be active if the enable bits are not set. |
| 2   | <b>BIOS_STS.</b> SMI# was generated due to ACPI software requesting attention (writing a 1 to the GBL_RLS bit with the BIOS_EN bit set). This bit is set by hardware and cleared by software writing a 1 to its bit position.        |
| 1:0 | Reserved.  |

### 8.8.3.13 IOMON\_STS\_EN — I/O Monitor Status and Enable Register

|                |             |            |             |
|----------------|-------------|------------|-------------|
| I/O Address:   | PMBASE +40h | Attribute: | R/W         |
| Default Value: | 0000h       | Size:      | 16 bits     |
| Lockable:      | No          | Usage:     | Legacy Only |
| Power Well:    | Core        |            |             |

| Bit   | Description  |
|-------|--|
| 15:14 | Reserved   |
| 13    | <b>IOMON2_STS.</b><br>1 = Indicates that the SMI# was caused by an access to the I/O monitor range set in the I/O Monitor Range 2 Register.<br>0 = Cleared by writing a 1 to the bit position. |
| 12    | <b>IOMON1_STS.</b><br>1 = Indicates that the SMI# was caused by an access to the I/O monitor range set in the I/O Monitor Range 1 Register<br>0 = Cleared by writing a 1 to the bit position.  |
| 11:10 | Reserved   |
| 9     | <b>IOMON2_EN.</b><br>1 = Enables the generation of an SMI# upon an access to the I/O monitor range set in the I/O Monitor Range 2 Register<br>0 = Disable.                                     |
| 8     | <b>IOMON1_EN.</b><br>1 = Enables the generation of an SMI# upon an access to the I/O monitor range set in the I/O Monitor Range 1 Register<br>0 = Disable                                      |
| 7:0   | Reserved   |

**8.8.3.14 DEVACT\_STS — Device Activity Status Register**

|                |             |            |             |
|----------------|-------------|------------|-------------|
| I/O Address:   | PMBASE +44h | Attribute: | R/W         |
| Default Value: | 0000h       | Size:      | 16 bits     |
| Lockable:      | No          | Usage:     | Legacy Only |
| Power Well:    | Core        |            |             |

A set bit, except for bits 6:9, indicates an access has been performed to the corresponding device's I/O range; for bits 6:9, a set bit indicates the corresponding PCI interrupt is active. Write 1 to the same bit position to clear it. This register is used in conjunction with the 1 minute SMI# timer to detect any system activity for legacy desktop power management.

| Bit | Description  |
|-----|--|
| 13  | <b>ADLIB_ACT_STA.</b> Ad-Lib. 1 = Enable. 0 = Disable.                                   |
| 12  | <b>KBC_ACT_STS.</b> KBC (60/64h). 1 = Enable. 0 = Disable.                               |
| 11  | <b>MIDI_ACT_STS.</b> MIDI. 1 = Enable. 0 = Disable.                                      |
| 10  | <b>AUDIO_ACT_STS.</b> Audio (Sound Blaster "OR'd" with MSS). 1 = Enable. 0 = Disable.    |
| 9   | <b>PIRQD_ACT_STS.</b> PIRQ[D]#. 1 = Enable. 0 = Disable.                                 |
| 8   | <b>PIRQC_ACT_STS.</b> PIRQ[C]#. 1 = Enable. 0 = Disable.                                 |
| 7   | <b>PIRQB_ACT_STS.</b> PIRQ[B]#. 1 = Enable. 0 = Disable.                                 |
| 6   | <b>PIRQA_ACT_STS.</b> PIRQ[A]#. 1 = Enable. 0 = Disable.                                 |
| 5   | <b>LEG_ACT_STS.</b> Paraller Port, Serial Port 1, Serial Port 2, Floppy Disk Controller. |
| 4   | Reserved.  |
| 3   | <b>IDES1_ACT_STS.</b> IDE Secondary Drive 1. 1 = Enable. 0 = Disable.                    |
| 2   | <b>IDES0_ACT_STS.</b> IDE Secondary Drive 0. 1 = Enable. 0 = Disable.                    |
| 1   | <b>IDEP1_ACT_STS.</b> IDE Primary Drive 1. 1 = Enable. 0 = Disable.                      |
| 0   | <b>IDEP0_ACT_STS.</b> IDE Primary Drive 0. 1 = Enable. 0 = Disable.                      |

**8.8.3.15 BUS\_ADDR\_TRACK— Bus Address Tracker**

|              |             |            |             |
|--------------|-------------|------------|-------------|
| I/O Address: | PMBASE +4Ch | Attribute: | RO          |
| Lockable:    | No          | Size:      | 16 bits     |
| Power Well:  | Core        | Usage:     | Legacy Only |

This register could be used by the SMI# handler to assist in determining what was the last cycle from the processor.

| Bit  | Description  |
|------|--|
| 15:0 | Corresponds to the low 16 bits of the last I/O cycle, as would be defined by the PCI AD[15:0] signals on the PCI bus (even though it may not be a real PCI cycle). The value is latched based on SMI# active. This functionality is useful for figuring out which I/O was last being accessed. |

### 8.8.3.16 BUS\_CYC\_TRACK— Bus Cycle Tracker

|              |             |            |             |
|--------------|-------------|------------|-------------|
| I/O Address: | PMBASE +4Eh | Attribute: | RO          |
| Lockable:    | No          | Size:      | 8 bits      |
| Power Well:  | Core        | Usage:     | Legacy Only |

This register could be used by the SMM handler to assist in determining what was the last cycle from the processor.

| Bit | Description   |
|-----|---|
| 7:4 | Corresponds to the byte enables, as would be defined by the PCI C/BE# signals on the PCI bus (even though it may not be a real PCI cycle). The value is latched based on SMI# going active. |
| 3:0 | Corresponds to the cycle type, as would be defined by the PCI C/BE# signals on the PCI bus (even though it may not be a real PCI cycle). The value is latched based on SMI# going active.   |

## 8.9 System Management TCO Registers (D31:F0)

The TCO logic is accessed via registers mapped to the PCI configuration space (Device 31:Function 0) and the system I/O space. For TCO PCI Configuration registers, see LPC Device 31:Function 0 PCI Configuration registers.

### 8.9.1 TCO Register I/O Map

The TCO I/O registers reside in a 32-byte range pointed to by a TCOBASE value, which is, ACPIBASE + 60h in the PCI config space. The following table shows the mapping of the registers within that 32-byte range. Each register is described in the sections below.

**Table 8-11. TCO I/O Register Map**

| Offset  | Read/Write | Register Name                               |
|---------|------------|---|
| 00h     | R/W        | TCO_RLD: TCO Timer Reload and Current Value |
| 01h     | R/W        | TCO_TMR: TCO Timer Initial Value            |
| 02h     | R/W        | TCO_DAT_IN: TCO Data In                     |
| 03h     | R/W        | TCO_DAT_OUT: TCO Data Out                   |
| 04h–05h | R/W        | TCO1_STS : TCO Status                       |
| 06h–07h | R/W        | TCO2_STS : TCO Status                       |
| 08h–09h | R/W        | TCO1_CNT: TCO Control                       |
| 0Ah–0Bh | R/W        | TCO2_CNT: TCO Control                       |

**NOTE:**

1. Reserved registers are read only and are not shown.

### 8.9.2 TCO1\_RLD—TCO Timer Reload and Current Value

I/O Address: TCOBASE +00h                      Attribute: R/W  
 Default Value: 0000h                            Size: 8 bits  
 Lockable: No                                        Power Well: Core

| Bit | Description   |
|-----|---|
| 7:0 | Reading this register will return the current count of the TCO timer. Writing any value to this register will reload the timer to prevent the timeout. Bits 7:6 will always be 0. |

### 8.9.3 TCO1\_TMR—TCO Timer Initial Value

I/O Address: TCOBASE +01h                      Attribute: R/W  
 Default Value: 0004h                            Size: 8 bits  
 Lockable: No                                        Power Well: Core

| Bit | Description   |
|-----|---|
| 7:6 | Reserved  |
| 5:0 | Value that is loaded into the timer each time the TCO_RLD register is written. Values of 0h–3h will be ignored and should not be attempted. The timer is clocked at approximately 0.6 seconds, and this allows timeouts ranging from 2.4 seconds to 38 seconds. |



### 8.9.4 TCO1\_DAT\_IN—TCO Data In Register

|                |              |             |        |
|----------------|--------------|-------------|--------|
| I/O Address:   | TCOBASE +02h | Attribute:  | R/W    |
| Default Value: | 0000h        | Size:       | 8 bits |
| Lockable:      | No           | Power Well: | Core   |

| Bit | Description  |
|-----|--|
| 7:0 | Data Register for passing commands from the OS to the SMI handler. Writes to this register will cause an SMI and set the OS_TCO_SMI bit in the TCO_STS register. |

### 8.9.5 TCO1\_DAT\_OUT—TCO Data Out Register

|                |              |             |        |
|----------------|--------------|-------------|--------|
| I/O Address:   | TCOBASE +03h | Attribute:  | R/W    |
| Default Value: | 0000h        | Size:       | 8 bits |
| Lockable:      | No           | Power Well: | Core   |

| Bit | Description  |
|-----|--|
| 7:0 | Data Register for passing commands from the SMI handler to the OS. Writes to this register will set the TCO_INT_STS bit in the TCO_STS register. It will also cause an interrupt, as selected by the TCO_INT_SEL bits. |

### 8.9.6 TCO1\_STS—TCO1 Status Register

|                |              |             |                                |
|----------------|--------------|-------------|--------------------------------|
| I/O Address:   | TCOBASE +04h | Attribute:  | R/W                            |
| Default Value: | 0000h        | Size:       | 16 bits                        |
| Lockable:      | No           | Power Well: | Core<br>(Except bit 7, in RTC) |

Unless otherwise indicated, these bits are all “sticky” and are cleared by writing a 1 to the corresponding bit position.

| Bit   | Description  |
|-------|--|
| 15:13 | Reserved   |
| 12    | <b>Host Controller SERR Status (HCSERR_STS).</b><br>1 = Host Controller indicates that it wants to cause an SERR#. The software must read the Host Controller to find out the reason for the Host Controller to generate the SERR#.<br>0 = Cleared by writing a 1 to its bit position. |
| 11    | <b>Host Controller NMI Status (HCNMI_STS).</b><br>1 = Host Controller indicates that it wants to cause an NMI. The software must read the Host Controller to find out the reason for the Host Controller to generate the NMI.<br>0 = Cleared by writing a 1 to its bit position.       |
| 10    | <b>Host Controller SMI Status (HCSMI_STS).</b><br>1 = Host Controller indicates that it wants to cause an SMI. The software must read the Host Controller to find out why it wanted the SMI.<br>0 = Cleared by writing a 1 to its bit position.  |
| 9     | <b>Host Controller SCI Status (HCSCI_STS).</b><br>1 = This bit is set to 1 if the Host Controller indicates that it wants to cause an SCI. The software must read the Host Controller to find out why it wanted the SCI.<br>0 = Cleared by writing a 1 to its bit position.            |

| Bit | Description  |
|-----|--|
| 8   | <p><b>BIOSWR_STS.</b></p> <p>1= ICH sets this bit and generates an SMI# to indicate an illegal attempt to write to the BIOS. This occurs when either: a) The BIOSWE bit is changed from 0 to 1 and the BLD bit is also set, or b) any write is attempted to the BIOS and the BIOSWE bit is also set.</p> <p><b>NOTE:</b> On write cycles attempted to the 4MB lower alias to the BIOS space, the BIOSWR_STS will not be set.</p> |
| 7   | <p><b>NEWCENTURY_STS.</b> This bit is in the RTC well.</p> <p>1 = This bit is set to 1 when the Year byte (RTC I/O space, index offset 09h) rolls over from 99 to 00. Setting this bit will cause an SMI# (but not a wake event).</p> <p>0 = Cleared by writing a 1 to the bit position or by RTCRST# going active.</p>  |
| 6:4 | Reserved   |
| 3   | <b>TIMEOUT.</b> Bit set to 1 by ICH to indicate that the SMI was caused by the TCO timer reaching 0.   |
| 2   | <p><b>TCO_INT_STS.</b></p> <p>1 = SMI handler caused the interrupt by writing to the TCO_DAT_OUT register.</p>   |
| 1   | <p><b>SW_TCO_SMI.</b></p> <p>1 = Software caused an SMI# by writing to the TCO_DAT_IN register.</p>  |
| 0   | <p><b>NMI2SMI_STS.</b></p> <p>1 = Set by the ICH when an SMI# occurs because an event occurred that would otherwise have caused an NMI.</p> <p>0 = Cleared by writing a 1 to this bit position.</p>  |

### 8.9.7 TCO2\_STS—TCO2 Status Register

|                |              |             |                                  |
|----------------|--------------|-------------|----------------------------------|
| I/O Address:   | TCOBASE +06h | Attribute:  | R/W                              |
| Default Value: | 0000h        | Size:       | 16 bits                          |
| Lockable:      | No           | Power Well: | Resume<br>(Except Bit 0, in RTC) |

| Bit  | Description   |
|------|---|
| 15:3 | Reserved  |
| 2    | <p><b>BOOT_STS.</b></p> <p>1 = Set to 1 when the SECOND_TO_STS bit goes from 0 to 1 and the processor has not fetched the first instruction.</p> <p>0 = Cleared by ICH based on RSMRST# or by software writing a 1 to this bit. Note that software should first clear the SECOND_TO_STS bit before writing a 1 to clear the BOOT_STS bit.</p> <p><b>NOTE:</b> If rebooting due to a second TCO timer timeout, and if the BOOT_STS bit is set, the ICH will reboot using the 'safe' multiplier (1111). This allows the system to recover from a processor frequency multiplier that is too high, and allows the BIOS to check the BOOT_STS bit at boot. If the bit is set and the frequency multiplier is 1111, then the BIOS knows that the processor has been programmed to an illegal multiplier.</p> |
| 1    | <p><b>SECOND_TO_STS.</b></p> <p>1 = The ICH sets this bit to a 1 to indicate that the TCO timer timed out a second time (probably due to system lock). If this bit is set the ICH will reboot the system after the second timeout. The reboot is done by asserting PCIRST#.</p> <p>0 = This bit is cleared by writing a 1 to the bit position or by a RSMRST#.</p>  |
| 0    | <p><b>Intruder Detect (INTRD_DET).</b> This bit is in the RTC well.</p> <p>1 = Set by ICH to indicate that an intrusion was detected. This bit is set even if the system is in G3 state.</p> <p>0 = This bit is only cleared by writing a 1 to the bit position, by RTCRST# or by a CF9h write.</p>   |

## 8.9.8 TCO1\_CNT—TCO1 Control Register

|                |              |             |         |
|----------------|--------------|-------------|---------|
| I/O Address:   | TCOBASE +08h | Attribute:  | R/W     |
| Default Value: | 0000h        | Size:       | 16 bits |
| Lockable:      | No           | Power Well: | Core    |

| Bit   | Description   |
|-------|---|
| 15:12 | Reserved  |
| 11    | <p><b>TCO Timer Halt.</b></p> <p>1 = The TCO Timer will halt. It will not count, and thus cannot reach a value that will cause an SMI# or set the SECOND_TO_STS bit.<br/>For the ICH (82801AA), when set, this bit will prevent rebooting and prevent Alert On LAN event messages (but not Alert On LAN heartbeat messages).</p> <p>0 = The TCO Timer is enabled to count.</p>  |
| 10    | <p><b>ICH (82801AA):</b></p> <p><b>SENDNOW.</b></p> <p>1 = Writing a 1 to this bit will cause the ICH to send an Alert On LAN Event message with the Software Event bit set.</p> <p>0 = The ICH will clear this bit to 0 when it has completed sending the message. Software must not set this bit to 1 again until the ICH has set it back to 0.</p> <p><b>ICH0 (82801AB):</b></p> <p>Reserved.</p>  |
| 9     | <p><b>NMI2SMI_EN.</b> Setting this bit to 1 while both NMI_EN bit and SMI_EN bit are set will force all NMI's to instead cause an SMI#. This will be reported in the TCO1_STS register. The SMI# handler can cause an NMI by writing a 1 to the NMI_NOW bit.</p> <p>When this bit is set to 1, no NMI will be generated. If this bit is set to 1, but the NMI_EN bit is not set, then no NMI or SMI# based on NMI events will be generated.</p> |
| 8     | <p><b>NMI_NOW.</b> Writing a 1 to this bit causes an NMI. This allows the BIOS or SMI handler to force an entry to the NMI handler. The NMI handler is expected to clear this bit. Another NMI will not be generated until the bit is cleared by writing back a 1.</p>  |
| 7:0   | Reserved  |

## 8.9.9 TCO2\_CNT—TCO2 Control Register

|                |              |             |         |
|----------------|--------------|-------------|---------|
| I/O Address:   | TCOBASE +0Ah | Attribute:  | R/W     |
| Default Value: | 0000h        | Size:       | 16 bits |
| Lockable:      | No           | Power Well: | Resume  |

| Bit  | Description  |
|------|--|
| 15:3 | Reserved.  |
| 2:1  | <p><b>INTRD_SEL.</b> Selects the action to take if the INTRUDER# signal goes active.</p> <p>00 = Reserved</p> <p>01 = Interrupt (as selected by TCO_INT_SEL).</p> <p>10 = SMI</p> <p>11 = Reserved</p> |
| 0    | Reserved.  |

### 8.9.10 TCO\_MESSAGE1 and TCO\_MESSAGE2 Registers (ICH 82801AA only)

|                |                          |             |        |
|----------------|--------------------------|-------------|--------|
| I/O Address:   | TCOBASE +0Ch (Message 1) | Attribute:  | R/W    |
|                | TCOBASE +0Dh (Message 2) |             |        |
| Default Value: | 00h                      | Size:       | 8 bits |
| Lockable:      | No                       | Power Well: | Resume |

Unless otherwise indicated, these bits are all “sticky” and are cleared by writing a 1 to the corresponding bit position.

| Bit | Description   |
|-----|---|
| 7:0 | The value written into this register will be sent out in the MESSAGE field of the Alert On LAN message. BIOS can write to this register to indicate its boot progress which can be monitored externally |

### 8.9.11 TCO\_WDSTATUS—TCO2 Control Register (ICH 82801AA only)

|                 |               |            |        |
|-----------------|---------------|------------|--------|
| Offset Address: | TCOBASE + 0Eh | Attribute: | R/W    |
| Default Value:  | 0000h         | Size:      | 8 bits |
| Power Well:     | Resume        |            |        |

| Bit | Description   |
|-----|---|
| 7:0 | <b>WDSTATUS.</b> The value written to this register will be sent in the Alert On LAN message. It can be used by the BIOS or system management software to indicate more details on the boot progress. |

## 8.10 General Purpose I/O Registers (D31:F0)

The control for the general purpose I/O signals is handled through a separate 64-byte I/O space. The base offset for this space is selected by the GPIO\_BAR register. Table 8-12 summarizes the ICH GPIO implementation.

Table 8-12. Summary of GPIO Implementation (Sheet 1 of 3)

| GPIO     | Type       | Alternate Function (1)  | Power Well | Notes   |
|----------|------------|---|------------|---|
| GPIO[0]  | Input Only | REQ[A]#   | Core       | <ul style="list-style-type: none"> <li>GPIO_USE_SEL bit 0 enables REQ/GNT[A]# pair.</li> <li>Input active status read from GPE1_STS register bit 0.</li> <li>Input active high/low set through GPI_INV register bit 0.</li> </ul> |
| GPIO[1]  | Input Only | REQ[B]# or REQ[5]# on ICH (82801AA)<br>REQ[B]# only on ICH0 (82801AB) | Core       | <ul style="list-style-type: none"> <li>GPIO_USE_SEL bit 1 enables REQ/GNT[B]# pair.</li> <li>Input active status read from GPE1_STS register bit 1.</li> <li>Input active high/low set through GPI_INV register bit 1.</li> </ul> |
| GPIO[2]  | N/A        | N/A   | N/A        | <ul style="list-style-type: none"> <li>Not Implemented</li> </ul>   |
| GPIO[3]  | N/A        | N/A   | N/A        | <ul style="list-style-type: none"> <li>Not Implemented</li> </ul>   |
| GPIO[4]  | N/A        | N/A   | N/A        | <ul style="list-style-type: none"> <li>Not Implemented</li> </ul>   |
| GPIO[5]  | Input Only | Unmuxed   | Core       | <ul style="list-style-type: none"> <li>Input active status read from GPE1_STS register bit 5.</li> <li>Input active high/low set through GPI_INV register bit 5.</li> </ul>   |
| GPIO[6]  | Input Only | Unmuxed   | Core       | <ul style="list-style-type: none"> <li>Input active status read from GPE1_STS register bit 6.</li> <li>Input active high/low set through GPI_INV register bit 6.</li> </ul>   |
| GPIO[7]  | Input Only | PERF# on ICH (82801AA)<br>Unmuxed on ICH0 (82801AB)                   | Core       | <ul style="list-style-type: none"> <li>Input active status read from GPE1_STS register bit 7.</li> <li>Input active high/low set through GPI_INV register bit 7.</li> </ul>   |
| GPIO[8]  | Input Only | LDRQ[1]#  | Resume     | <ul style="list-style-type: none"> <li>Input active status read from GPE1_STS register bit 8.</li> <li>Input active high/low set through GPI_INV register bit 8.</li> </ul>   |
| GPIO[9]  | Input Only | AC_SDIN[1]  | Resume     | <ul style="list-style-type: none"> <li>Input active status read from GPE1_STS register bit 9.</li> <li>Input active high/low set through GPI_INV register bit 9.</li> </ul>   |
| GPIO[10] | N/A        | N/A   | N/A        | <ul style="list-style-type: none"> <li>Not Implemented</li> </ul>   |
| GPIO[11] | Input Only | SMBALERT#   | Resume     | <ul style="list-style-type: none"> <li>Input active status read from GPE1_STS register bit 11.</li> <li>Input active high/low set through GPI_INV register bit 11.</li> </ul>   |

Table 8-12. Summary of GPIO Implementation (Sheet 2 of 3)

| GPIO     | Type        | Alternate Function (1)  | Power Well | Notes  |
|----------|-------------|---|------------|--|
| GPIO[12] | Input Only  | Unmuxed   | Resume     | <ul style="list-style-type: none"> <li>Input active status read from GPE1_STS register bit 12.</li> <li>Input active high/low set through GPI_INV register bit 12.</li> </ul>  |
| GPIO[13] | Input Only  | Unmuxed   | Resume     | <ul style="list-style-type: none"> <li>Input active status read from GPE1_STS register bit 13.</li> <li>Input active high/low set through GPI_INV register bit 13.</li> </ul>  |
| GPIO[14] | N/A         | N/A   | N/A        | <ul style="list-style-type: none"> <li>Not Implemented</li> </ul>  |
| GPIO[15] | N/A         | N/A   | N/A        | <ul style="list-style-type: none"> <li>Not Implemented</li> </ul>  |
| GPIO[16] | Output Only | GNT[A]#   | Core       | <ul style="list-style-type: none"> <li>Blink enabled via GPO_BLINK register bit 16.</li> <li>Output controlled via GP_LVL register bit 16.</li> <li>TTL driver output</li> </ul>   |
| GPIO[17] | Output Only | GNT[B]# or GNT[5]# on ICH (82801AA)<br>GNT[B]# only on ICH0 (82801AB) | Core       | <ul style="list-style-type: none"> <li>Blink enabled via GPO_BLINK register bit 17.</li> <li>Output controlled via GP_LVL register bit 17.</li> <li>TTL driver output</li> </ul>   |
| GPIO[18] | N/A         | N/A   | N/A        | <ul style="list-style-type: none"> <li>Not Implemented</li> </ul>  |
| GPIO[19] | N/A         | N/A   | N/A        | <ul style="list-style-type: none"> <li>Not Implemented</li> </ul>  |
| GPIO[20] | N/A         | N/A   | N/A        | <ul style="list-style-type: none"> <li>Not Implemented</li> </ul>  |
| GPIO[21] | Output Only | Unmuxed   | Core       | <ul style="list-style-type: none"> <li>Blink enabled via GPO_BLINK register bit 21.</li> <li>This GPIO defaults high for connection to MISA NOGO input.</li> <li>Output controlled via GP_LVL register bit 21.</li> <li>TTL driver output</li> </ul> |
| GPIO[22] | Output Only | Unmuxed   | Core       | <ul style="list-style-type: none"> <li>Blink enabled via GPO_BLINK register bit 22.</li> <li>Output controlled via GP_LVL register bit 22.</li> <li>TTL driver output</li> </ul>   |
| GPIO[23] | Output Only | Unmuxed   | Core       | <ul style="list-style-type: none"> <li>Blink enabled via GPO_BLINK register bit 23.</li> <li>Output controlled via GP_LVL register bit 23.</li> <li>Open drain driver output</li> </ul>  |
| GPIO[24] | Output Only | SLP_S3#   | Resume     | <ul style="list-style-type: none"> <li>Blink enabled via GPO_BLINK register bit 24.</li> <li>Output controlled via GP_LVL register bit 24.</li> <li>TTL driver output</li> </ul>   |
| GPIO[25] | Output Only | SUS_STAT#   | Resume     | <ul style="list-style-type: none"> <li>Blink enabled via GPO_BLINK register bit 25.</li> <li>Output controlled via GP_LVL register bit 25.</li> <li>TTL driver output</li> </ul>   |
| GPIO[26] | Output Only | SUS_CLK on ICH (82801AA)<br>Unmuxed on ICH0 (82801AB)                 | Resume     | <ul style="list-style-type: none"> <li>Output controlled via GP_LVL register bit 26.</li> <li>Input active status read from GP_LVL register bit 26.</li> <li>TTL driver output</li> </ul>  |

Table 8-12. Summary of GPIO Implementation (Sheet 3 of 3)

| GPIO     | Type           | Alternate Function (1)                                  | Power Well | Notes  |
|----------|----------------|---|------------|--|
| GPIO[27] | Input / Output | ALERTCLK on ICH (82801AA)<br>Unmuxed on ICH0 (82801AB)  | Resume     | For the ICH (82801AA), GPIO_USE_SEL bit 27 enables ALERTCLK/ALERTDATA pair. <ul style="list-style-type: none"> <li>• Blink enabled via GPO_BLINK register bit 27.</li> <li>• Output controlled via GP_LVL register bit 27.</li> <li>• Input active status read from GP_LVL register bit 27.</li> <li>• Input active high/low set through GPI_INV register bit 27.</li> <li>• Input / Output select set via GP_IO_SEL register bit 27.</li> <li>• Open drain driver output</li> </ul> |
| GPIO[28] | Input / Output | ALERTDATA on ICH (82801AA)<br>Unmuxed on ICH0 (82801AB) | Resume     | For the ICH (82801AA), GPIO_USE_SEL bit 28 enables ALERTCLK/ALERTDATA pair. <ul style="list-style-type: none"> <li>• Blink enabled via GPO_BLINK register bit 28.</li> <li>• Output controlled via GP_LVL register bit 28.</li> <li>• Input active high/low set through GPI_INV register bit 28.</li> <li>• Input / Output select set via GP_IO_SEL register bit 28.</li> <li>• Open Drain driver output</li> </ul>  |
| GPIO[29] | N/A            | N/A   | N/A        | • Not Implemented  |
| GPIO[30] | N/A            | N/A   | N/A        | • Not Implemented  |
| GPIO[31] | N/A            | N/A   | N/A        | • Not Implemented  |

**NOTES:**

1. All GPIOs default to their alternate function.
2. All inputs are sticky. The status bit will remain set as long as the input was asserted for 2 clocks. GPIOs are sampled on PCI clocks in S0/S1. GPIOs are sampled on RTC clocks in S3/S4/S5.
3. All GPIOs can be routed to cause an SCI or SMI#

### 8.10.1 GPIO Register I/O Address Map

Table 8-13. Registers to Control GPIO

| Offset                          | Mnemonic     | Register Name                  | Default   | Access |
|---------------------------------|--------------|--------------------------------|---|--------|
| <b>General Registers</b>        |              |                                |   |        |
| 00–03h                          | GPIO_USE_SEL | GPIO Use Select                | 00E03060h<br>(ICH: 82801AA<br>0CE030E0<br>(ICH0: 82801AB) | R/W    |
| 04–07h                          | GP_IO_SEL    | GPIO Input/Output Select       | 0000FFFFh   | R/W    |
| 08–0Bh                          |              | Reserved                       | 0   | RO     |
| 0C–0Fh                          | GP_LVL       | GPIO Level for Input or Output | 00200000h   | R/W    |
| 10–13h                          |              | Reserved                       | 0   | RO     |
| <b>Output Control Registers</b> |              |                                |   |        |
| 14–17h                          | GPO_TTL      | GPIO TTL Select                | 06630000h   | RO     |
| 18–1Bh                          | GPO_BLINK    | GPIO Blink Enable              | 00000000h   | R/W    |
| 1C–1Fh                          |              | Reserved                       | 0   | RO     |
| <b>Input Control Registers</b>  |              |                                |   |        |
| 20–2Bh                          |              | Reserved                       | 00000000h   | RO     |
| 2C–2Fh                          | GPI_INV      | GPIO Signal Invert             | 00000000h   | R/W    |



## 8.10.2 GPIO\_USE\_SEL—GPIO Use Select Register

|                 |   |             |         |
|-----------------|---|-------------|---------|
| Offset Address: | GPIOBASE + 00h  | Attribute:  | R/W     |
| Lockable:       | Yes   | Size:       | 32 bits |
|                 |   | Power Well: | Resume  |
| Default Value:  | 00E03060h (ICH: 82801AA)<br>0CE030E0h (ICH0: 82801AB) |             |         |

For GPIOx signal pins that are multiplexed with other functions, the bits in this register select either the GPIOx or alternative function.

| Bit   | Description  |
|-------|--|
| 31:28 | Reserved (HW =0).  |
| 27    | <b>ICH (82801AA):</b><br><b>GPIO_USE_SEL.</b><br>1 = GPIO[28] and GPIO[27].<br>0 = ALERTDATA AND ALERTCLK<br><b>ICH0 (82801AB):</b><br>Reserved (HW =1). |
| 26    | <b>ICH (82801AA):</b><br><b>GPIO_USE_SEL.</b><br>1 = GPIO[26]<br>0 = SUS_CLK<br><b>ICH0 (82801AB):</b><br>Reserved (HW =1).                              |
| 25    | <b>GPIO_USE_SEL.</b><br>1 = GPIO[25]<br>0 = SUS_STAT#  |
| 24    | <b>GPIO_USE_SEL.</b><br>1 = GPIO[24]<br>0 = SLP_S3#  |
| 23:21 | Reserved (HW=1).   |
| 20:14 | Reserved (HW=0).   |
| 13:12 | Reserved (HW=1)  |
| 11    | <b>GPIO_USE_SEL.</b><br>1 = GPIO[11]<br>0 = SMBALERT#  |
| 10    | Reserved (HW=0).   |
| 9     | <b>GPIO_USE_SEL.</b><br>1 = GPIO[9]<br>0 = AC_SDIN[1]  |
| 8     | <b>GPIO_USE_SEL.</b><br>1 = GPIO[8]<br>0 = LDRQ[1]#  |

| Bit | Description  |
|-----|--|
| 7   | <b>ICH (82801AA):</b><br><b>GPIO_USE_SEL.</b><br>1 = GPIO[7]<br>0 = PERR#<br><b>ICH0 (82801AB):</b><br>Reserved (HW=1)   |
| 6:5 | Reserved (HW=1)  |
| 4:2 | Reserved (HW=0)  |
| 1   | <b>ICH (82801AA):</b><br><b>GPIO_USE_SEL.</b><br>1 = GPIO[1] and GPIO[17]<br>0 = GNT[B]#/REQ[B]# or GNT[5]#/REQ[5]#<br><b>ICH0 (82801AB):</b><br><b>GPIO_USE_SEL.</b><br>1 = GPIO[1] and GPIO[17]<br>0 = GNT[B]#/REQ[B]# |
| 0   | <b>GPIO_USE_SEL.</b><br>1 = GPIO[1] and GPIO[16]<br>0 = GNT[A]#/REQ[A]#  |

### 8.10.3 GP\_IO\_SEL—GPIO Input/Output Select Register

|                |               |             |         |
|----------------|---------------|-------------|---------|
| Offset         | GPIOBASE +04h | Attribute:  | R/W     |
| Default Value: | 0000FFFFh     | Size:       | 32 bits |
| Lockable:      | No            | Power Well: | Resume  |

For the enabled input/output GPIOx signals (enabled via the GPIO\_USE\_SEL register), the bits in this register select between input and output.

| Bit   | Description  |
|-------|--|
| 31:29 | Reserved (HW=0).   |
| 28    | <b>GP_IO_SEL.</b><br>1 = Input (GPIO[28]).<br>0 = Output (GPIO[28]). |
| 27    | <b>GP_IO_SEL.</b><br>1 = Input (GPIO[27]).<br>0 = Output (GPIO[27]). |
| 21:26 | Reserved (HW=0).   |
| 20:18 | Reserved.  |
| 17:16 | Reserved (HW=0).   |
| 15:14 | Reserved.  |
| 13:5  | Reserved (HW=1).   |
| 4:2   | Reserved.  |
| 1:0   | Reserved (HW=1).   |

### 8.10.4 GP\_LVL—GPIO Level for Input or Output Register

|                |               |             |         |
|----------------|---------------|-------------|---------|
| Offset         | GPIOBASE +0Ch | Attribute:  | R/W     |
| Default Value: | 00200000h     | Size:       | 32 bits |
| Lockable:      | No            | Power Well: | Resume  |

| Bit   | Description  |
|-------|--|
| 31:29 | Reserved. Will always be 0. Writes have no effect.   |
| 28:21 | <p><b>GP_LVL.</b> Bits 28:21 correspond to GPIO[28:21], respectively.</p> <p><b>GPIO Programmed as Output.</b> If a GPIO is programmed to be an output (via the corresponding IO_SEL bit), then the corresponding GP_LVL bit can be updated by software to drive a high or low value on the output pin.</p> <p>1 = High (corresponding GPIO signal)<br/>0 = Low (corresponding GPIO signal)</p> <p><b>GPIO Programmed as Input.</b> If a GPIO is programmed as an input, and if the corresponding GPI_INV bit is 0, then the corresponding GP_LVL bit reflects the state of the input signal. Writes to GP_LVL bits for GPIO set up as inputs will have no effect.</p> <p>1 = High (corresponding GPIO signal)<br/>0 = Low (corresponding GPIO signal)</p> |
| 20:18 | Reserved. Will always be 0. Writes have no effect.   |
| 17:16 | <b>GP_LVL.</b> Bits 17:16 correspond to GPIO[17:16], respectively. See Bits [28:21] for functional description.  |
| 15:0  | Reserved. Will always be 0. Writes have no effect.   |

**NOTES:**

- For GPI[1:0], [13:5], the active status of a GPI is read from the corresponding bit in GPE1\_STS register.
- Bit 21 defaults to 1 to allow GPIO[21] to be connected to the NOGO signal of the 82380AB PCI-TO-ISA Bridge.

### 8.10.5 GPO\_TTL—GPIO TTL Select Register

|                |               |             |         |
|----------------|---------------|-------------|---------|
| Offset         | GPIOBASE +14h | Attribute:  | RO      |
| Default Value: | 06630000h     | Size:       | 32 bits |
| Lockable:      | No            | Power Well: | Resume  |

When a bit is set to 1, and the corresponding GP\_IO\_SEL bit is programmed as an output, the output signal is configured as a standard high/low totem-pole driver. The output will be driven high when the corresponding GP\_LVL bit is 1 and driven low when the corresponding GP\_LVL bit is 0.

When a bit is 0, and the corresponding GP\_IO\_SEL bit is programmed as an output, the output signal will be an open-drain style driver: tri-state when the corresponding GP\_LVL bit is 1 and driven low with the corresponding GP\_LVL bit is 0.

| Bit   | Description  |
|-------|--|
| 31:29 | Reserved. Always 0.  |
| 28:27 | Always '0'. These GPIOs are open-drain outputs. Writes have no effect. |
| 26:24 | Always '1'. These GPIOs are TTL outputs. Writes have no effect.        |
| 23    | Always '0'. These GPIOs are open-drain outputs. Writes have no effect. |
| 22:21 | Always '1'. These GPIOs are TTL outputs. Writes have no effect.        |
| 20:18 | Reserved. Always 0.  |
| 17:16 | Always '1'. These GPIOs are TTL outputs. Writes have no effect.        |
| 15:0  | Reserved. Always 0.  |

### 8.10.6 GPO\_BLINK—GPO Blink Enable Register

Offset: GPIOBASE +18h      Attribute: R/W  
 Default Value: 0000h      Size: 32 bits  
 Lockable: No      Power Well: Resume

| Bit   | Description   |
|-------|---|
| 31:29 | Reserved  |
| 28:27 | <b>GP_BLINK.</b> Bits 28:27 correspond to GPIO[28:27], respectively. The setting of this bit has no effect if the corresponding GPIO signal is programmed as an input.<br>0 = The corresponding GPIO signal will function normally.<br>1 = When 1, and the corresponding GPIO is programmed as an output, the output signal will blink at a rate of approximately once per second. The high and low times are approximately 50%.<br>The value of the corresponding GP_LVL bit remains unchanged during the blink process, and does not effect the blink in any way. |
| 26    | Reserved  |
| 25:21 | <b>GP_BLINK.</b> Bits 25:21 correspond to GPIO[25:21], respectively. See bits [28:27] description.  |
| 20:18 | Reserved  |
| 17:16 | <b>GP_BLINK.</b> Bits 17:16 correspond to GPIO[17:16], respectively. See bits [28:27] description.  |
| 15:0  | Reserved  |

### 8.10.7 GPI\_INV—GPIO Signal Invert Register

Offset: GPIOBASE +2Ch      Attribute: R/W  
 Default Value: 00000000h      Size: 32 bits  
 Lockable: No      Power Well: Resume

| Bit   | Description  |
|-------|--|
| 31:29 | Reserved.  |
| 28:27 | <b>Input Inversion.</b> These bits correspond to GPIO[28:27], respectively. The bit only has effect if the corresponding GPIO is used as an input. For inputs, if the Inversion bit is set, then the data value in the GP_LVL bit is inverted. This is used to allow active-low and active-high inputs to cause SMI# or SCI. |
| 26:14 | Reserved.  |
| 13:5  | <b>Input Inversion.</b> These bits correspond to GPIO[13:5], respectively. See Bits [28:27] description.   |
| 4:2   | Reserved.  |
| 1:0   | <b>Input Inversion.</b> These bits correspond to GPIO[1:0], respectively. See Bits [28:27] description.  |

# IDE Controller Registers (D31:F1)

# 9

## 9.1 PCI Configuration Registers (IDE—D31:F1)

*Note:* All of the IDE registers are in the Core well. None can be locked.

Table 9-1. PCI Configuration Map (IDE-D31:F1)

| Offset  | Mnemonic   | Register Name/Function           | Default   | Type |
|---------|------------|----------------------------------|---|------|
| 00h–01h | VID        | Vendor ID                        | 8086h   | RO   |
| 02h–03h | DID        | Device ID                        | 2411h<br>(ICH: 82801AA)<br>2421h<br>(ICH0: 82801AB) | RO   |
| 04h–05h | CMD        | Command Register                 | 00h   | R/W  |
| 06h–07h | STS        | Device Status                    | 0280h   | R/W  |
| 08h     | RID        | Revision ID                      | Note 1  | RO   |
| 09h     | PI         | Programming Interface            | 80h   | RO   |
| 0Ah     | SCC        | Sub Class Code                   | 01h   | RO   |
| 0Bh     | BCC        | Base Class Code                  | 01h   | RO   |
| 0Dh     | MLT        | Master Latency Timer             | 00  | RO   |
| 0Eh     | HTYPE      | Header Type                      | 00h   | RO   |
| 20h–23h | BAR        | Base Address Register            | 00000001h   | R/W  |
| 40h–43h | IDETIMP    | Primary IDE Timing               | 0000h   | R/W  |
| 44h     | SIDETIM    | Slave IDE Timing                 | 00h   | R/W  |
| 48h     | SDMAC      | Synchronous DMA Control Register | 00h   | R/W  |
| 4Ah–4Bh | SDMATIM    | Synchronous DMA Timing Register  | 0000h   | R/W  |
| 54h     | IDE_CONFIG | IDE I/O Configuration Register   | 00h   | R/W  |

**NOTES:**

1. Refer to ICH Specification Updates for the value of the Revision ID.
2. Addresses that are not listed in the above table are reserved and read only.
3. The ICH IDE controller is not arbitrated as a PCI device, therefore it does not need a master latency timer.

### 9.1.1 VID—Vendor ID Register (IDE—D31:F1)

Offset Address: 00–01h                      Attribute: RO  
 Default Value: 8086h                      Size: 16 bits

| Bit  | Description   |
|------|---|
| 15:0 | <b>Vendor ID Value.</b> This is a 16 bit value assigned to Intel. Intel VID = 8086h |

### 9.1.2 DID—Device ID Register (IDE—D31:F1)

Offset Address: 02–03h                      Attribute: RO  
 Size: 16 bits  
 Default Value: 2411h (ICH: 82801AA)  
 2421h (ICH0: 82901AB)

| Bit  | Description  |
|------|--|
| 15:0 | <b>Device ID Value.</b> This is a 16 bit value assigned to the ICH IDE Bridge. |

### 9.1.3 CMD—Command Register (IDE—D31:F1)

Address Offset: 04h–05h                      Attribute: R/W  
 Default Value: 00h                      Size: 16 bits

| Bit   | Description  |
|-------|--|
| 15:10 | Reserved.  |
| 9     | Fast Back to Back Enable (FBE)—RO: Reserved as '0'.  |
| 8     | SERR# Enable—RO: Reserved as '0'.  |
| 7     | Wait Cycle Control—RO: Reserved as '0'.  |
| 6     | Parity Error Response—RO: Reserved as '0'.   |
| 5     | VGA Palette Snoop—RO: Reserved as '0'.   |
| 4     | Postable Memory Write Enable (PMWE)—RO: Reserved as '0'.   |
| 3     | Special Cycle Enable (SCE)—RO: Reserved as '0'.  |
| 2     | <b>Bus Master Enable (BME)—R/W.</b> Controls the ICH's ability to act as a PCI master for IDE Bus Master transfers.<br>1 = Enable<br>0 = Disable   |
| 1     | Memory Space Enable (MSE)—RO: Reserved as '0'.   |
| 0     | <b>I/O Space Enable (IOSE).</b> This bit controls access to the I/O space registers.<br>1 = Enable. Access to the Legacy IDE ports (both Primary and Secondary) and the Bus Master IO registers is enabled. The Base Address register for the Bus Master registers should be programmed before this bit is set.<br>0 = Disable.<br><br><b>NOTE:</b> Separate bits are provided to independently disable the Primary or Secondary I/O spaces if implemented externally. |









| Bit | Description   |
|-----|---|
| 7   | <b>Drive 1 DMA Timing Enable (DTE1).</b><br>1 = Enable the fast timing mode for DMA transfers only for this drive. PIO transfers to the IDE data port will run in compatible timing.<br>0 = Disable   |
| 6   | <b>Drive 1 Prefetch/Posting Enable (PPE1).</b><br>1 = Enable Prefetch and posting to the IDE data port for this drive.<br>0 = Disable   |
| 5   | <b>Drive 1 IORDY Sample Point Enable (IE1).</b><br>1 = Enable IORDY sampling for this drive.<br>0 = Disable IORDY sampling for this drive.  |
| 4   | <b>Drive 1 Fast Timing Bank (TIME1).</b><br>0 = Accesses to the data port will use compatible timings for this drive.<br>1 = When set and bit 14 cleared, accesses to the data port will use bits 13:12 for the IORDY sample point, and bits 9:8 for the recovery time. When set and bit 14 set, accesses to the data port will use the IORDY sample point and recover time specified in the slave IDE timing register. |
| 3   | <b>Drive 0 DMA Timing Enable (DTE0).</b><br>1 = Enable fast timing mode for DMA transfers only for this drive. PIO transfers to the IDE data port will run in compatible timing.<br>0 = Disable   |
| 2   | <b>Drive 0 Prefetch/Posting Enable (PPE0).</b><br>1 = Enable prefetch and posting to the IDE data port for this drive.<br>0 = Disable   |
| 1   | <b>Drive 0 IORDY Sample Point Enable (IE0).</b><br>1 = Enable IORDY sampling for this drive.<br>0 = Disable IORDY sampling is disabled for this drive.  |
| 0   | <b>Drive 0 Fast Timing Bank (TIME0).</b><br>0 = Accesses to the data port will use compatible timings for this drive.<br>1 = Accesses to the data port will use bits 13:12 for the IORDY sample point, and bits 9:8 for the recovery time   |

### 9.1.13 SLV\_IDETIM—Slave (Drive 1) IDE Timing Register (IDE—D31:F1)

Address Offset: 44h Attribute: R/W  
 Default Value: 00h Size: 8 bits

| Bit | Description  |
|-----|--|
| 7:6 | <b>Secondary Drive 1 IORDY Sample Point (SISP1)</b> . The setting of these bits determines the number of PCI clocks between IDE IOR#/IOW# assertion and the first IORDY sample point, if the access is to drive 1 data port and bit 14 of the IDE timing register for secondary is set.<br>00 = 5 clocks<br>01 = 4 clocks<br>10 = 3 clocks<br>11 = Reserved                  |
| 5:4 | <b>Secondary Drive 1 Recovery Time (SRCT1)</b> . The setting of these bits determines the minimum number of PCI clocks between the last IORDY sample point and the IOR#/IOW# strobe of the next cycle, if the access is to drive 1 data port and bit 14 of the IDE timing register for secondary is set.<br>00 = 4 clocks<br>01 = 3 clocks<br>10 = 2 clocks<br>11 = 1 clocks |
| 3:2 | <b>Primary Drive 1 IORDY Sample Point (PISP1)</b> . The setting of these bits determines the number of PCI clocks between IOR#/IOW# assertion and the first IORDY sample point, if the access is to drive 1 data port and bit 14 of the IDE timing register for primary is set.<br>00 = 5 clocks<br>01 = 4 clocks<br>10 = 3 clocks<br>11 = Reserved                          |
| 1:0 | <b>Primary Drive 1 Recovery Time (PRCT1)</b> . The setting of these bits determines the minimum number of PCI clocks between the last IORDY sample point and the IOR#/IOW# strobe of the next cycle, if the access is to drive 1 data port and bit 14 of the IDE timing register for primary is set.<br>00 = 4 clocks<br>01 = 3 clocks<br>10 = 2 clocks<br>11 = 1 clocks     |

### 9.1.14 SDMA\_CNT—Synchronous DMA Control Register (IDE—D31:F1)

Address Offset: 48h  
Default Value: 00h

Attribute: R/W  
Size: 8 bits

| Bit | Description   |
|-----|---|
| 7:4 | Reserved.   |
| 3   | <b>Secondary Drive 1 Synchronous DMA Mode Enable (SSDE1).</b><br>1 = Enable Synchronous DMA mode for secondary channel drive 1<br>0 = Disable (default) |
| 2   | <b>Secondary Drive 0 Synchronous DMA Mode Enable (SSDE0).</b><br>1 = Enable Synchronous DMA mode for secondary drive 0<br>0 = Disable (default).        |
| 1   | <b>Primary Drive 1 Synchronous DMA Mode Enable (PSDE1).</b><br>1 = Enable Synchronous DMA mode for primary channel drive 1<br>0 = Disable (default)     |
| 0   | <b>Primary Drive 0 Synchronous DMA Mode Enable (PSDE0).</b><br>1 = Enable Synchronous DMA mode for primary channel drive 0<br>0 = Disable (default)     |

### 9.1.15 SDMA\_TIM—Synchronous DMA Timing Register (IDE—D31:F1)

Address Offset: 4A–4Bh  
 Default Value: 0000h

Attribute: R/W  
 Size: 16 bits

| Bit                            | Description   |                                |   |                               |               |                               |                               |                               |                               |               |               |
|--------------------------------|---|--------------------------------|---|-------------------------------|---------------|-------------------------------|-------------------------------|-------------------------------|-------------------------------|---------------|---------------|
| 15:14                          | Reserved.   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 13:12                          | <p><b>Secondary Drive 1 Cycle Time (SCT1).</b> For Sync DMA mode. The setting of these bits determines the minimum write strobe cycle time (CT). The DMARDY#-to-STOP (RP) time is also determined by the setting of these bits.</p> <table border="0"> <tr> <td><b>SCB1 = 0 (33 MHz clock)</b></td> <td><b>SCB1 = 1 (66 MHz clock) (ICH 82801AA only)</b></td> </tr> <tr> <td>00 = CT 4 clocks, RP 6 clocks</td> <td>00 = Reserved</td> </tr> <tr> <td>01 = CT 3 clocks, RP 5 clocks</td> <td>01 = CT 3 clocks, RP 8 clocks</td> </tr> <tr> <td>10 = CT 2 clocks, RP 4 clocks</td> <td>10 = CT 2 clocks, RP 8 clocks</td> </tr> <tr> <td>11 = Reserved</td> <td>11 = Reserved</td> </tr> </table> | <b>SCB1 = 0 (33 MHz clock)</b> | <b>SCB1 = 1 (66 MHz clock) (ICH 82801AA only)</b> | 00 = CT 4 clocks, RP 6 clocks | 00 = Reserved | 01 = CT 3 clocks, RP 5 clocks | 01 = CT 3 clocks, RP 8 clocks | 10 = CT 2 clocks, RP 4 clocks | 10 = CT 2 clocks, RP 8 clocks | 11 = Reserved | 11 = Reserved |
| <b>SCB1 = 0 (33 MHz clock)</b> | <b>SCB1 = 1 (66 MHz clock) (ICH 82801AA only)</b>   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 00 = CT 4 clocks, RP 6 clocks  | 00 = Reserved   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 01 = CT 3 clocks, RP 5 clocks  | 01 = CT 3 clocks, RP 8 clocks   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 10 = CT 2 clocks, RP 4 clocks  | 10 = CT 2 clocks, RP 8 clocks   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 11 = Reserved                  | 11 = Reserved   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 11:10                          | Reserved.   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 9:8                            | <p><b>Secondary Drive 0 Cycle Time (SCT0).</b> For Sync DMA mode. The setting of these bits determines the minimum write strobe cycle time (CT). The DMARDY#-to-STOP (RP) time is also determined by the setting of these bits.</p> <table border="0"> <tr> <td><b>SCB0 = 0 (33 MHz clock)</b></td> <td><b>SCB0 = 1 (66 MHz clock) (ICH 82801AA only)</b></td> </tr> <tr> <td>00 = CT 4 clocks, RP 6 clocks</td> <td>00 = Reserved</td> </tr> <tr> <td>01 = CT 3 clocks, RP 5 clocks</td> <td>01 = CT 3 clocks, RP 8 clocks</td> </tr> <tr> <td>10 = CT 2 clocks, RP 4 clocks</td> <td>10 = CT 2 clocks, RP 8 clocks</td> </tr> <tr> <td>11 = Reserved</td> <td>11 = Reserved</td> </tr> </table> | <b>SCB0 = 0 (33 MHz clock)</b> | <b>SCB0 = 1 (66 MHz clock) (ICH 82801AA only)</b> | 00 = CT 4 clocks, RP 6 clocks | 00 = Reserved | 01 = CT 3 clocks, RP 5 clocks | 01 = CT 3 clocks, RP 8 clocks | 10 = CT 2 clocks, RP 4 clocks | 10 = CT 2 clocks, RP 8 clocks | 11 = Reserved | 11 = Reserved |
| <b>SCB0 = 0 (33 MHz clock)</b> | <b>SCB0 = 1 (66 MHz clock) (ICH 82801AA only)</b>   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 00 = CT 4 clocks, RP 6 clocks  | 00 = Reserved   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 01 = CT 3 clocks, RP 5 clocks  | 01 = CT 3 clocks, RP 8 clocks   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 10 = CT 2 clocks, RP 4 clocks  | 10 = CT 2 clocks, RP 8 clocks   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 11 = Reserved                  | 11 = Reserved   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 7:6                            | Reserved.   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 5:4                            | <p><b>Primary Drive 1 Cycle Time (PCT1).</b> For Sync DMA mode, the setting of these bits determines the minimum write strobe cycle time(CT). The DMARDY#-to-STOP (RP) time is also determined by the setting of these bits.</p> <table border="0"> <tr> <td><b>PCB1 = 0 (33 MHz clock)</b></td> <td><b>PCB1 = 1 (66 MHz clock) (ICH 82801AA only)</b></td> </tr> <tr> <td>00 = CT 4 clocks, RP 6 clocks</td> <td>00 = Reserved</td> </tr> <tr> <td>01 = CT 3 clocks, RP 5 clocks</td> <td>01 = CT 3 clocks, RP 8 clocks</td> </tr> <tr> <td>10 = CT 2 clocks, RP 4 clocks</td> <td>10 = CT 2 clocks, RP 8 clocks</td> </tr> <tr> <td>11 = Reserved</td> <td>11 = Reserved</td> </tr> </table>    | <b>PCB1 = 0 (33 MHz clock)</b> | <b>PCB1 = 1 (66 MHz clock) (ICH 82801AA only)</b> | 00 = CT 4 clocks, RP 6 clocks | 00 = Reserved | 01 = CT 3 clocks, RP 5 clocks | 01 = CT 3 clocks, RP 8 clocks | 10 = CT 2 clocks, RP 4 clocks | 10 = CT 2 clocks, RP 8 clocks | 11 = Reserved | 11 = Reserved |
| <b>PCB1 = 0 (33 MHz clock)</b> | <b>PCB1 = 1 (66 MHz clock) (ICH 82801AA only)</b>   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 00 = CT 4 clocks, RP 6 clocks  | 00 = Reserved   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 01 = CT 3 clocks, RP 5 clocks  | 01 = CT 3 clocks, RP 8 clocks   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 10 = CT 2 clocks, RP 4 clocks  | 10 = CT 2 clocks, RP 8 clocks   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 11 = Reserved                  | 11 = Reserved   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 3:2                            | Reserved.   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 1:0                            | <p><b>Primary Drive 0 Cycle Time (PCT0).</b> For Sync DMA mode, the setting of these bits determines the minimum write strobe cycle time (CT). The DMARDY#-to-STOP (RP) time is also determined by the setting of these bits.</p> <table border="0"> <tr> <td><b>PCB0 = 0 (33 MHz clock)</b></td> <td><b>PCB0 = 1 (66 MHz clock) (ICH 82801AA only)</b></td> </tr> <tr> <td>00 = CT 4 clocks, RP 6 clocks</td> <td>00 = Reserved</td> </tr> <tr> <td>01 = CT 3 clocks, RP 5 clocks</td> <td>01 = CT 3 clocks, RP 8 clocks</td> </tr> <tr> <td>10 = CT 2 clocks, RP 4 clocks</td> <td>10 = CT 2 clocks, RP 8 clocks</td> </tr> <tr> <td>11 = Reserved</td> <td>11 = Reserved</td> </tr> </table>   | <b>PCB0 = 0 (33 MHz clock)</b> | <b>PCB0 = 1 (66 MHz clock) (ICH 82801AA only)</b> | 00 = CT 4 clocks, RP 6 clocks | 00 = Reserved | 01 = CT 3 clocks, RP 5 clocks | 01 = CT 3 clocks, RP 8 clocks | 10 = CT 2 clocks, RP 4 clocks | 10 = CT 2 clocks, RP 8 clocks | 11 = Reserved | 11 = Reserved |
| <b>PCB0 = 0 (33 MHz clock)</b> | <b>PCB0 = 1 (66 MHz clock) (ICH 82801AA only)</b>   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 00 = CT 4 clocks, RP 6 clocks  | 00 = Reserved   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 01 = CT 3 clocks, RP 5 clocks  | 01 = CT 3 clocks, RP 8 clocks   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 10 = CT 2 clocks, RP 4 clocks  | 10 = CT 2 clocks, RP 8 clocks   |                                |   |                               |               |                               |                               |                               |                               |               |               |
| 11 = Reserved                  | 11 = Reserved   |                                |   |                               |               |                               |                               |                               |                               |               |               |



| Bit | Description   |
|-----|---|
| 0   | <p><b>ICH (82801AA):</b><br/> <b>PCB0: Primary Drive 0 Base Clock</b><br/>           1 = 66 MHz base clock for UDMA timings<br/>           0 = 33 MHz base clock for UDMA timings</p> <p><b>ICH0 (82801AB):</b><br/>           Reserved. Always '0'. Writes have no effect.</p> |









# USB Controller Registers (D31:F2) 10

## 10.1 PCI Configuration Registers (D31:F2)

Table 10-1. PCI Configuration Map (USB—D31:F2)

| Offset | Mnemonic   | Register Name/Function            | Function 2 Default                                  | Type |
|--------|------------|-----------------------------------|---|------|
| 00–01h | VID        | Vendor ID                         | 8086h   | RO   |
| 02–03h | DID        | Device ID                         | 2412h<br>(ICH: 82801AA)<br>2422h<br>(ICH0: 82801AB) | RO   |
| 04–05h | CMD        | Command Register                  | 0000h   | R/W  |
| 06–07h | STA        | Device Status                     | 0280h   | R/W  |
| 08h    | RID        | Revision ID                       | Note 1  | RO   |
| 09h    | PI         | Programming Interface             | 00h   | RO   |
| 0Ah    | SCC        | Sub Class Code                    | 03h   | RO   |
| 0Bh    | BCC        | Base Class Code                   | 0Ch   | RO   |
| 0Eh    | HTYPE      | Header Type                       | 00h   | RO   |
| 20–23h | Base       | Base Address Register             | 00000001h   | R/W  |
| 3Ch    | INTR_LN    | Interrupt Line                    | 00h   | R/W  |
| 3Dh    | INTR_PN    | Interrupt Pin                     | 04h   | RO   |
| 60h    | SB_RELNUM  | Serial Bus Release Number         | 10h   | RO   |
| C0–C1h | USB_LEGKEY | USB Legacy Keyboard/Mouse Control | 2000h   | R/W  |
| C4h    | USB_RES    | USB Resume Enable                 | 00h   | R/W  |

**NOTES:**

1. Refer to ICH Specification Updates for the value of the Revision ID.
2. Addresses that are not listed in the above table are reserved and read only.



### 10.1.4 STA—Device Status Register (USB—D31:F2)

Address Offset: 06–07h                      Attribute: R/W  
 Default Value: 0280h                      Size: 16 bits

| Bit  | Description  |
|------|--|
| 15   | Reserved as '0'. Read Only.  |
| 14   | Reserved as '0'. Read Only.  |
| 13   | <b>Received Master-Abort Status (RMA)—R/WC.</b><br>1 = USB, as a master, generated a master-abort.<br>0 = Software sets RMA to 0 by writing a 1 to this bit location.  |
| 12   | Reserved. Always read as '0'.  |
| 11   | <b>Signaled Target-Abort Status (STA)—R/WC.</b><br>1 = USB function is targeted with a transaction that the ICH terminates with a target abort.<br>0 = Software resets STA to 0 by writing a 1 to this bit location..                          |
| 10:9 | <b>DEVSEL# Timing Status (DEVT)—RO.</b> This 2-bit field defines the timing for DEVSEL# assertion. These read only bits indicate the ICH's DEVSEL# timing when performing a positive decode. ICH generates DEVSEL# with medium timing for USB. |
| 8    | Data Parity Error Detected—RO: Reserved as '0'.  |
| 7    | Fast Back-to-Back Capable—RO: Reserved as '1'  |
| 6    | User Definable Features (UDF)—RO: Reserved as '0'.   |
| 5    | 66 MHz Capable—RO: Reserved as '0'.  |
| 4:0  | Reserved.  |

### 10.1.5 RID—Revision Identification Register (USB—D31:F2)

Address Offset: 08h                              Attribute: RO  
 Default Value: See ICH Spec. Updates      Size: 8 bits

| Bit | Description   |
|-----|---|
| 7:0 | These bits contain device stepping information and are hardwired to the default value. Refer to ICH Specification Updates for this value. |

### 10.1.6 PI—Programming Interface (USB—D31:F2)

Address Offset: 09h                              Attribute: RO  
 Default Value: 00h                              Size: 8 bits

| Bit | Description   |
|-----|---|
| 7:0 | Programming Interface.<br>00h = No specific register level programming interface defined. |





### 10.1.14 USB\_LEGKEY—USB Legacy Keyboard/Mouse Control Register (USB—D31:F2)

Address Offset: C0–C1      Attribute: R/W  
 Default Value: 2000h      Size: 16 bits

| Bit | Description  |
|-----|--|
| 15  | <b>SMI Caused by End of Pass-through (SMIBYENDPS).</b> Indicates if the event occurred. Note that even if the corresponding enable bit is not set in the Bit 0, then this bit will still be active. It is up to the SMM code to use the enable bit to determine the exact cause of the SMI#.<br>1 = Event Occurred<br>0 = Cleared by writing a 1 to it.  |
| 14  | Reserved.  |
| 13  | <b>PCI Interrupt Enable (USBPIRQEN).</b> Used to prevent the USB controller from generating an interrupt due to transactions on its ports. Note that it will probably be configured to generate an SMI using bit 4 of this register. Default to 1 for compatibility with older USB software.<br>1 = Enable<br>0 = Disable  |
| 12  | <b>SMI Caused by USB Interrupt (SMIBYUSB).</b> Indicates if the event occurred. Note that even if the corresponding enable bit is not set in the Bit 4, then this bit will still be active. It is up to the SMM code to use the enable bit to determine the exact cause of the SMI#.<br>1 = Event Occurred<br>0 = Software should clear the IRQ via the USB controller. Writing a 1 to this bit will have no effect. |
| 11  | <b>SMI Caused by Port 64 Write (TRAPBY64W).</b> Indicates if the event occurred. Note that even if the corresponding enable bit is not set in the Bit 3, then this bit will still be active. It is up to the SMM code to use the enable bit to determine the exact cause of the SMI#.<br>1 = Event Occurred<br>0 = Cleared by writing a 1 to it.   |
| 10  | <b>SMI Caused by Port 64 Read (TRAPBY64R).</b> Indicates if the event occurred. Note that even if the corresponding enable bit is not set in the Bit 2, then this bit will still be active. It is up to the SMM code to use the enable bit to determine the exact cause of the SMI#.<br>1 = Event Occurred<br>0 = Cleared by writing a 1 to it.  |
| 9   | <b>SMI Caused by Port 60 Write (TRAPBY60W).</b> Indicates if the event occurred. Note that even if the corresponding enable bit is not set in the Bit 1, then this bit will still be active. It is up to the SMM code to use the enable bit to determine the exact cause of the SMI#.<br>1 = Event Occurred<br>0 = Cleared by writing a 1 to it.   |
| 8   | <b>SMI Caused by Port 60 Read (TRAPBY60R).</b> Indicates if the event occurred. Note that even if the corresponding enable bit is not set in the Bit 0, then this bit will still be active. It is up to the SMM code to use the enable bit to determine the exact cause of the SMI#.<br>1 = Event Occurred<br>0 = Cleared by writing a 1 to it.  |
| 7   | <b>SMI at End of Pass-through Enable (SMIATENDPS).</b> May need to cause SMI at the end of a pass-through. Can occur if an SMI is generated in the middle of a pass through, and needs to be serviced later.   |
| 6   | <b>Pass Through State (PSTATE).</b><br>1 = Indicates that the state machine is in the middle of an A20GATE pass-through sequence.<br>0 = If software needs to reset this bit, it should set Bit 5 to 0.  |



| Bit | Description  |
|-----|--|
| 5   | <b>A20Gate Pass-Through Enable (A20PASSEN).</b><br>1 = Enable. Allows A20GATE sequence Pass-Through function (see details below). SMI# will not be generated, even if the various enable bits are set.<br>0 = Disable. |
| 4   | <b>SMI on USB IRQ (USBSMIEN).</b><br>1 = USB interrupt will cause an SMI event.  |
| 3   | <b>SMI on Port 64 Writes Enable (64WEN).</b><br>1 = Write to port 64h will cause an SMI event.   |
| 2   | <b>SMI on Port 64 Reads Enable (64REN).</b><br>1 = Read to port 64h will cause an SMI event.   |
| 1   | <b>SMI on Port 60 Writes Enable (60WEN).</b><br>1 = Write to port 60h will cause an SMI event.   |
| 0   | <b>SMI on Port 60 Reads Enable (60REN).</b><br>1 = Read to port 60h will cause an SMI event.   |

### 10.1.15 USB\_RES—USB Resume Enable Register (USB—D31:F2)

Address Offset: C4h                      Attribute: R/W  
 Default Value: 00h                      Size: 8 bits

| Bit | Description   |
|-----|---|
| 7:2 | Reserved.   |
| 1   | <b>PORT1EN.</b> Enable port 1 of the USB controller to look at wakeup events.<br>1 = The USB controller will monitor port 1 for remote wakeup and connect/disconnect events.<br>0 = The USB controller will not look at this port for a wakeup event. |
| 0   | <b>PORT0EN.</b> Enable port 0 of the USB controller to look at wakeup events.<br>1 = the USB controller will monitor port 0 for remote wakeup and connect/disconnect events.<br>0 = The USB controller will not look at this port for a wakeup event. |

## 10.2 USB I/O Registers

Some of the read/write register bits which deal with changing the state of the USB hub ports function such that on read back they reflect the current state of the port and not necessarily the state of the last write to the register. This allows the software to poll the state of the port and wait until it is in the proper state before proceeding. A Host Controller Reset, Global Reset, or Port Reset will immediately terminate a transfer on the affected ports and disable the port. This affects the USBCMD register, bit [4] and the PORTSC registers, bits [12,6,2]. See individual bit descriptions for more detail.

Table 10-2. USB I/O Registers

| Offset  | Mnemonic  | Register                    | Default   | Type              |
|---------|-----------|-----------------------------|-----------|-------------------|
| 00h–01h | USBCMD    | USB Command Register        | 0000h     | R/W               |
| 02h–03h | USBSTS    | USB Status Register         | 0000h     | R/WC              |
| 04h–05h | USBINTR   | USB Interrupt Enable        | 0000h     | R/W               |
| 06h–07h | FRNUM     | USB Frame Number            | 0000h     | R/W*              |
| 08h–0Bh | FRBASEADD | USB Frame List Base Address | Undefined | R/W               |
| 0Ch     | SOFMOD    | USB Start of Frame Modify   | 40h       | R/W               |
| 0Dh–0Fh |           | Reserved                    | 0         | RO                |
| 10h–11h | PORTSC0   | Port 0 Status/Control       | 0080h     | R/WC <sup>1</sup> |
| 12h–13h | PORTSC1   | Port 1 Status/Control       | 0080h     | R/WC <sup>1</sup> |
| 14h–17h |           | Reserved                    | 0         | RO                |
| 18h     | LOOPDATA  | Loop Back Test Data         | 00h       | RO                |

**NOTES:**

1. These registers are WORD writable only. Byte writes to these registers have unpredictable effects.

### 10.2.1 USBCMD—USB Command Register

I/O Offset:                   Base + (00–01h)                   Attribute:                   R/W  
 Default Value:               0000h                               Size:                        16 bits

The Command Register indicates the command to be executed by the serial bus host controller. Writing to the register causes a command to be executed. The table following the bit description provides additional information on the operation of the Run/Stop and Debug bits.

| Bit  | Description   |
|------|---|
| 15:7 | Reserved..  |
| 8    | <b>Loop Back Test Mode.</b><br>1 = ICH is in loop back test mode. When both ports are connected together, a write to one port will be seen on the other port and the data will be stored in I/O offset 18h.<br>0 = Disable loop back test mode. |

| Bit | Description  |
|-----|--|
| 7   | <p><b>Max Packet (MAXP).</b> This bit selects the maximum packet size that can be used for full speed bandwidth reclamation at the end of a frame. This value is used by the Host Controller to determine whether it should initiate another transaction based on the time remaining in the SOF counter. Use of reclamation packets larger than the programmed size will cause a Babble error if executed during the critical window at frame end. The Babble error results in the offending endpoint being stalled. Software is responsible for ensuring that any packet which could be executed under bandwidth reclamation be within this size limit.</p> <p>1 = 64 bytes<br/>0 = 32 bytes</p>  |
| 6   | <p><b>Configure Flag (CF).</b> HCD software sets this bit as the last action in its process of configuring the Host Controller. This bit has no effect on the hardware. It is provided only as a semaphore service for software.</p>   |
| 5   | <p><b>Software Debug (SWDBG).</b> The SWDBG bit must only be manipulated when the controller is in the stopped state. This can be determined by checking the HCHalted bit in the USBSTS register.</p> <p>1 = Debug mode. In SW Debug mode, the Host Controller clears the Run/Stop bit after the completion of each USB transaction. The next transaction is executed when software sets the Run/Stop bit back to 1.</p> <p>0 = Normal Mode.</p>   |
| 4   | <p><b>Force Global Resume (FGR).</b></p> <p>1 = Host Controller sends the Global Resume signal on the USB. The Host Controller sets this bit to 1 when a resume event (connect, disconnect, or K-state) is detected while in global suspend mode.</p> <p>0 = Software sets this bit to 0 after 20 ms has elapsed to stop sending the Global Resume signal. At that time all USB devices should be ready for bus activity.</p> <p>Software resets this bit to 0 to end Global Resume signaling. The 1 to 0 transition causes the port to send a low speed EOP signal. This bit will remain a 1 until the EOP has completed.</p>   |
| 3   | <p><b>Enter Global Suspend Mode (EGSM).</b></p> <p>1 = Host Controller enters the Global Suspend mode. No USB transactions occurs during this time. The Host Controller is able to receive resume signals from USB and interrupt the system. Software must ensure that the Run/Stop bit (bit 0) is cleared prior to setting this bit.</p> <p>0 = Software resets this bit to 0 to come out of Global Suspend mode. Software writes this bit to 0 at the same time that Force Global Resume (bit 4) is written to 0 or after writing bit 4 to 0.</p>  |
| 2   | <p><b>Global Reset (GRESET).</b></p> <p>1 = Global Reset. The Host Controller sends the global reset signal on the USB and then resets all its logic, including the internal hub registers. The hub registers are reset to their power on state. Chip Hardware Reset has the same effect as Global Reset (bit 2), except that the Host Controller does not send the Global Reset on USB.</p> <p>0 = This bit is reset by the software after a minimum of 10 ms has elapsed as specified in Chapter 7 of the USB Specification.</p>   |
| 1   | <p><b>Host Controller Reset (HCRESET).</b> The HCRESET effects on Hub registers are slightly different from Chip Hardware Reset and Global USB Reset. The HCRESET affects bits [8,3:0] of the Port Status and Control Register (PORTSC) of each port. HCRESET resets the state machines of the Host Controller including the Connect/Disconnect state machine (one for each port). When the Connect/Disconnect state machine is reset, the output that signals connect/disconnect are negated to 0, effectively signaling a disconnect, even if a device is attached to the port. This virtual disconnect causes the port to be disabled. This disconnect and disabling of the port causes bit 1 (connect status change) and bit 3 (port enable/disable change) of the PORTSC to get set. The disconnect also causes bit 8 of PORTSC to reset. About 64 bit times after HCRESET goes to 0, the connect and low-speed detect will take place and bits 0 and 8 of the PORTSC will change accordingly.</p> <p>1 = Reset. When this bit is set, the Host Controller module resets its internal timers, counters, state machines, etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. This bit is reset by the Host Controller when the reset process is complete.</p> |

| Bit | Description   |
|-----|---|
| 0   | <p><b>Run/Stop (RS).</b> When set to 1, the ICH proceeds with execution of the schedule. The ICH continues execution as long as this bit is set. When this bit is cleared, the ICH completes the current transaction on the USB and then halts. The HC Halted bit in the status register indicates when the Host Controller has finished the transaction and has entered the stopped state. The Host Controller clears this bit when the following fatal errors occur: consistency check failure, PCI Bus errors.</p> <p>1 = Run<br/>0 = Stop</p> |

**Table 10-3. Run/Stop, Debug Bit Interaction SWDBG (Bit 5), Run/Stop (Bit 0) Operation**

| SWDBG (Bit 5) | Run/Stop (Bit 0) | Description  |
|---------------|------------------|--|
| 0             | 0                | If executing a command, the Host Controller completes the command and then stops. The 1.0 ms frame counter is reset and command list execution resumes from start of frame using the frame list pointer selected by the current value in the FRNUM register. (While Run/Stop=0, the FRNUM register can be reprogrammed).                                       |
| 0             | 1                | Execution of the command list resumes from Start Of Frame using the frame list pointer selected by the current value in the FRNUM register. The Host Controller remains running until the Run/Stop bit is cleared (by Software or Hardware).   |
| 1             | 0                | If executing a command, the Host Controller completes the command and then stops and the 1.0 ms frame counter is frozen at its current value. All status are preserved. The Host Controller begins execution of the command list from where it left off when the Run/Stop bit is set.  |
| 1             | 1                | Execution of the command list resumes from where the previous execution stopped. The Run/Stop bit is set to 0 by the Host Controller when a TD is being fetched. This causes the Host Controller to stop again after the execution of the TD (single step). When the Host Controller has completed execution, the HC Halted bit in the Status Register is set. |

When the USB Host Controller is in Software Debug Mode (USBCMD Register bit 5=1), the single stepping software debug operation is as follows:

To Enter Software Debug Mode:

1. HCD puts Host Controller in Stop state by setting the Run/Stop bit to 0.
2. HCD puts Host Controller in Debug Mode by setting the SWDBG bit to 1.
3. HCD sets up the correct command list and Start Of Frame value for starting point in the Frame List Single Step Loop.
4. HCD sets Run/Stop bit to 1.
5. Host Controller executes next active TD, sets Run/Stop bit to 0, and stops.
6. HCD reads the USBCMD register to check if the single step execution is completed (HCHalted=1).
7. HCD checks results of TD execution. Go to step 4 to execute next TD or step 8 to end Software Debug mode.
8. HCD ends Software Debug mode by setting SWDBG bit to 0.
9. HCD sets up normal command list and Frame List table.
10. HCD sets Run/Stop bit to 1 to resume normal schedule execution.

In Software Debug mode, when the Run/Stop bit is set, the Host Controller starts. When a valid TD is found, the Run/Stop bit is reset. When the TD is finished, the HCHalted bit in the USBSTS register (bit 5) is set.

The SW Debug mode skips over inactive TDs and only halts after an active TD has been executed. When the last active TD in a frame has been executed, the Host Controller waits until the next SOF is sent and then fetches the first TD of the next frame before halting.

This HCHalted bit can also be used outside of Software Debug mode to indicate when the Host Controller has detected the Run/Stop bit and has completed the current transaction. Outside of the Software Debug mode, setting the Run/Stop bit to 0 always resets the SOF counter so that when the Run/Stop bit is set the Host Controller starts over again from the frame list location pointed to by the Frame List Index (see FRNUM Register description) rather than continuing where it stopped.

## 10.2.2 USBSTA—USB Status Register

|                |                 |            |                   |
|----------------|-----------------|------------|-------------------|
| I/O Offset:    | Base + (02–03h) | Attribute: | Read/ Write Clear |
| Default Value: | 0000h           | Size:      | 16 bits           |

This register indicates pending interrupts and various states of the Host Controller. The status resulting from a transaction on the serial bus is not indicated in this register. Software sets a bit to 0 in this register by writing a 1 to it.

| Bit  | Description  |
|------|--|
| 15:6 | Reserved.  |
| 5    | <b>HCHalted.</b> The Host Controller sets this bit to 1 after it has stopped executing as a result of the Run/Stop bit being set to 0, either by software or by the Host Controller hardware (debug mode or an internal error).  |
| 4    | <b>Host Controller Process Error.</b> The Host Controller sets this bit to 1 when it detects a fatal error and indicates that the Host Controller suffered a consistency check failure while processing a Transfer Descriptor. An example of a consistency check failure would be finding an illegal PID field while processing the packet header portion of the TD. When this error occurs, the Host Controller clears the Run/Stop bit in the Command register to prevent further schedule execution. A hardware interrupt is generated to the system. |
| 3    | <b>Host System Error.</b> The Host Controller sets this bit to 1 when a serious error occurs during a host system access involving the Host Controller module. In a PCI system, conditions that set this bit to 1 include PCI Parity error, PCI Master Abort, and PCI Target Abort. When this error occurs, the Host Controller clears the Run/Stop bit in the Command register to prevent further execution of the scheduled TDs. A hardware interrupt is generated to the system.  |
| 2    | <b>Resume Detect (RSM_DET).</b> The Host Controller sets this bit to 1 when it receives a “RESUME” signal from a USB device. This is only valid if the Host Controller is in a global suspend state (bit 3 of Command register = 1).   |
| 1    | <b>USB Error Interrupt.</b> The Host Controller sets this bit to 1 when completion of a USB transaction results in an error condition (e.g., error counter underflow). If the TD on which the error interrupt occurred also had its IOC bit set, both this bit and Bit 0 are set.  |
| 0    | <b>USB Interrupt (USBINT).</b> The Host Controller sets this bit to 1 when the cause of an interrupt is a completion of a USB transaction whose Transfer Descriptor had its IOC bit set. The Host Controller also sets this bit to 1 when a short packet is detected (actual length field in TD is less than maximum length field in TD), and short packet detection is enabled in that TD.  |

### 10.2.3 USBINTR—Interrupt Enable Register

I/O Offset: Base + (04–05h) Attribute: R/W  
 Default Value: 0000h Size: 16 bits

This register enables and disables reporting of the corresponding interrupt to the software. When a bit is set and the corresponding interrupt is active, an interrupt is generated to the host. Fatal errors (Host Controller Processor Error-bit 4, USBSTS Register) cannot be disabled by the host controller. Interrupt sources that are disabled in this register still appear in the Status Register to allow the software to poll for events.

| Bit  | Description   |
|------|---|
| 15:4 | Reserved.   |
| 3    | <b>Short Packet Interrupt Enable.</b><br>1 = Enabled.<br>0 = Disabled.      |
| 2    | <b>Interrupt On Complete (IOC) Enable.</b><br>1 = Enabled.<br>0 = Disabled. |
| 1    | <b>Resume Interrupt Enable.</b><br>1 = Enabled.<br>0 = Disabled.            |
| 0    | <b>Timeout/CRC Interrupt Enable.</b><br>1 = Enabled.<br>0 = Disabled.       |

### 10.2.4 FRNUM—Frame Number Register

I/O Offset: Base + (06–07h) Attribute: R/W (Writes must be Word Writes)  
 Default Value: 0000h Size: 16 bits

Bits [10:0] of this register contain the current frame number which is included in the frame SOF packet. This register reflects the count value of the internal frame number counter. Bits [9:0] are used to select a particular entry in the Frame List during scheduled execution. This register is updated at the end of each frame time.

This register must be written as a word. Byte writes are not supported. This register cannot be written unless the Host Controller is in the STOPPED state as indicated by the HCHalted bit (USBSTS register). A write to this register while the Run/Stop bit is set (USBCMD register) is ignored.

| Bit   | Description   |
|-------|---|
| 15:11 | Reserved.   |
| 10:0  | <b>Frame List Current Index/Frame Number.</b> Bits [10:0] provide the frame number in the SOF Frame. The value in this register increments at the end of each time frame (approximately every 1 ms). In addition, bits [9:0] are used for the Frame List current index and correspond to memory address signals [11:2]. |

### 10.2.5 FRBASEADD—Frame List Base Address

I/O Offset: Base + (08–0Bh) Attribute: R/W  
 Default Value: Undefined Size: 32 bits

This 32-bit register contains the beginning address of the Frame List in the system memory. HCD loads this register prior to starting the schedule execution by the Host Controller. When written, only the upper 20 bits are used. The lower 12 bits are written as zero (4-Kbyte alignment). The contents of this register are combined with the frame number counter to enable the Host Controller to step through the Frame List in sequence. The two least significant bits are always 00. This requires DWord alignment for all list entries. This configuration supports 1024 Frame List entries.

| Bit   | Description   |
|-------|---|
| 31:12 | <b>Base Address.</b> These bits correspond to memory address signals [31:12], respectively. |
| 11:0  | Reserved.   |

### 10.2.6 SOFMOD—Start of Frame Modify Register

I/O Offset: Base + (0Ch) Attribute: R/W  
 Default Value: 40h Size: 8 bits

This 1-byte register is used to modify the value used in the generation of SOF timing on the USB. Only the 7 least significant bits are used. When a new value is written into these 7 bits, the SOF timing of the next frame will be adjusted. This feature can be used to adjust out any offset from the clock source that generates the clock that drives the SOF counter. This register can also be used to maintain real time synchronization with the rest of the system so that all devices have the same sense of real time. Using this register, the frame length can be adjusted across the full range required by the USB specification. Its initial programmed value is system dependent based on the accuracy of hardware USB clock and is initialized by system BIOS. It may be reprogrammed by USB system software at any time. Its value will take effect from the beginning of the next frame. This register is reset upon a Host Controller Reset or Global Reset. Software must maintain a copy of its value for reprogramming if necessary.

| Bit   | Description   |   |                             |       |   |       |   |   |   |   |   |       |    |       |    |       |    |   |   |   |   |       |     |       |     |
|---|---|---|-----------------------------|-------|---|-------|---|---|---|---|---|-------|----|-------|----|-------|----|---|---|---|---|-------|-----|-------|-----|
| 7   | Reserved.   |   |                             |       |   |       |   |   |   |   |   |       |    |       |    |       |    |   |   |   |   |       |     |       |     |
| 6:0   | <p><b>SOF Timing Value.</b> Guidelines for the modification of frame time are contained in Chapter 7 of the USB Specification. The SOF cycle time (number of SOF counter clock periods to generate a SOF frame length) is equal to 11936 + value in this field. The default value is decimal 64 which gives a SOF cycle time of 12000. For a 12 MHz SOF counter clock input, this produces a 1 ms Frame period. The following table indicates what SOF Timing Value to program into this field for a certain frame period.</p> <table border="1"> <thead> <tr> <th>Frame Length<br/>(# 12Mhz Clocks)<br/>(decimal)</th> <th>SOF Reg. Value<br/>(decimal)</th> </tr> </thead> <tbody> <tr><td>11936</td><td>0</td></tr> <tr><td>11937</td><td>1</td></tr> <tr><td>.</td><td>.</td></tr> <tr><td>.</td><td>.</td></tr> <tr><td>11999</td><td>63</td></tr> <tr><td>12000</td><td>64</td></tr> <tr><td>12001</td><td>65</td></tr> <tr><td>.</td><td>.</td></tr> <tr><td>.</td><td>.</td></tr> <tr><td>12062</td><td>126</td></tr> <tr><td>12063</td><td>127</td></tr> </tbody> </table> | Frame Length<br>(# 12Mhz Clocks)<br>(decimal) | SOF Reg. Value<br>(decimal) | 11936 | 0 | 11937 | 1 | . | . | . | . | 11999 | 63 | 12000 | 64 | 12001 | 65 | . | . | . | . | 12062 | 126 | 12063 | 127 |
| Frame Length<br>(# 12Mhz Clocks)<br>(decimal) | SOF Reg. Value<br>(decimal)   |   |                             |       |   |       |   |   |   |   |   |       |    |       |    |       |    |   |   |   |   |       |     |       |     |
| 11936   | 0   |   |                             |       |   |       |   |   |   |   |   |       |    |       |    |       |    |   |   |   |   |       |     |       |     |
| 11937   | 1   |   |                             |       |   |       |   |   |   |   |   |       |    |       |    |       |    |   |   |   |   |       |     |       |     |
| .   | .   |   |                             |       |   |       |   |   |   |   |   |       |    |       |    |       |    |   |   |   |   |       |     |       |     |
| .   | .   |   |                             |       |   |       |   |   |   |   |   |       |    |       |    |       |    |   |   |   |   |       |     |       |     |
| 11999   | 63  |   |                             |       |   |       |   |   |   |   |   |       |    |       |    |       |    |   |   |   |   |       |     |       |     |
| 12000   | 64  |   |                             |       |   |       |   |   |   |   |   |       |    |       |    |       |    |   |   |   |   |       |     |       |     |
| 12001   | 65  |   |                             |       |   |       |   |   |   |   |   |       |    |       |    |       |    |   |   |   |   |       |     |       |     |
| .   | .   |   |                             |       |   |       |   |   |   |   |   |       |    |       |    |       |    |   |   |   |   |       |     |       |     |
| .   | .   |   |                             |       |   |       |   |   |   |   |   |       |    |       |    |       |    |   |   |   |   |       |     |       |     |
| 12062   | 126   |   |                             |       |   |       |   |   |   |   |   |       |    |       |    |       |    |   |   |   |   |       |     |       |     |
| 12063   | 127   |   |                             |       |   |       |   |   |   |   |   |       |    |       |    |       |    |   |   |   |   |       |     |       |     |

### 10.2.7 PORTSC[0,1]—Port Status and Control Register

I/O Offset: Port 0: Base + (10–11h) Attribute: R/W (Writes must be a Word write)  
 Port 1: Base + (12–13h)  
 Default Value: 0080h Size: 16 bits

After a Power-up reset, Global reset, or Host Controller reset, the initial conditions of a port are: no device connected, Port disabled, and the bus line status is 00 (single-ended zero).

| Bit         | Description  |             |           |     |         |     |        |     |         |
|-------------|--|-------------|-----------|-----|---------|-----|--------|-----|---------|
| 15:13       | Reserved— RO.  |             |           |     |         |     |        |     |         |
| 12          | <p><b>Suspend</b>— R/W. This bit should not be written to a 1 if global suspend is active (bit 3=1 in the USBCMD register). Bit 2 and bit 12 of this register define the hub states as follows:</p> <table border="1"> <thead> <tr> <th>Bits [12,2]</th> <th>Hub State</th> </tr> </thead> <tbody> <tr><td>x,0</td><td>Disable</td></tr> <tr><td>0,1</td><td>Enable</td></tr> <tr><td>1,1</td><td>Suspend</td></tr> </tbody> </table> <p>When in suspend state, downstream propagation of data is blocked on this port, except for single-ended 0 resets (global reset and port reset). The blocking occurs at the end of the current transaction, if a transaction was in progress when this bit was written to 1. In the suspend state, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB.</p> <p>1 = Port in suspend state.<br/>                     0 = Port not in suspend state.</p> <p>Note: Suspending a port is supposed to stop when the current transaction completes. However, if there is a specific error condition, the ICH may issue a start-of-frame, and then shut down.</p> | Bits [12,2] | Hub State | x,0 | Disable | 0,1 | Enable | 1,1 | Suspend |
| Bits [12,2] | Hub State  |             |           |     |         |     |        |     |         |
| x,0         | Disable  |             |           |     |         |     |        |     |         |
| 0,1         | Enable   |             |           |     |         |     |        |     |         |
| 1,1         | Suspend  |             |           |     |         |     |        |     |         |
| 11          | <p><b>Overcurrent Indicator</b>— R/WC. Set by hardware</p> <p>1 = Overcurrent pin has gone from inactive to active on this port.<br/>                     0 = This bit is cleared by software writing a '1'.</p>   |             |           |     |         |     |        |     |         |



| Bit | Description   |
|-----|---|
| 10  | <b>Overcurrent Active</b> — RO.<br>1 = Set by hardware to indicate that the overcurrent pin is active (low).<br>0 = Cleared by hardware to indicate that the overcurrent pin is inactive (high).  |
| 9   | <b>Port Reset</b> — RO. When set, the port is disabled and sends the USB Reset signaling.   |
| 8   | <b>Low Speed Device Attached (LS)</b> — RO. Writes have no effect.<br>1 = Low speed device is attached to this port.<br>0 = Full speed device is attached.  |
| 7   | Reserved— RO: Always read as 1.   |
| 6   | <b>Resume Detect (RSM_DET)</b> — R/W. Software sets this bit to a 1 to drive resume signaling. The Host Controller sets this bit to a 1 if a J-to-K transition is detected for at least 32 microseconds while the port is in the Suspend state. The ICH will then reflect the K-state back onto the bus as long as the bit remains a '1', and the port is still in the suspend state (bit 12,2 are '11'). Writing a 0 (from 1) causes the port to send a low speed EOP. This bit will remain a 1 until the EOP has completed.<br>1 = Resume detected/driven on port.<br>0 = No resume (K-state) detected/driven on port.  |
| 5:4 | <b>Line Status</b> — RO. These bits reflect the D+ (bit 4) and D- (bit 5) signals lines' logical levels. These bits are used for fault detect and recovery as well as for USB diagnostics. This field is updated at EOF2 time.  |
| 3   | <b>Port Enable/Disable Change</b> — RW. For the root hub, this bit gets set only when a port is disabled due to disconnect on that port or due to the appropriate conditions existing at the EOF2 point (See Chapter 11 of the USB Specification). Software clears this bit by writing a 1 to it.<br>1 = Port enabled/disabled status has changed.<br>0 = No change.  |
| 2   | <b>Port Enabled/Disabled (PORT_EN)</b> — R/W. Ports can be enabled by host software only. Ports can be disabled by either a fault condition (disconnect event or other fault condition) or by host software. Note that the bit status does not change until the port state actually changes and that there may be a delay in disabling or enabling a port if there is a transaction currently in progress on the USB.<br>1 = Enable.<br>0 = Disable.  |
| 1   | <b>Connect Status Change</b> — RW. Indicates that a change has occurred in the port's Current Connect Status (see bit 0). The hub device sets this bit for any changes to the port device connect status, even if system software has not cleared a connect status change. If, for example, the insertion status changes twice before system software has cleared the changed condition, hub hardware will be setting an already-set bit (i.e., the bit will remain set). However, the hub transfers the change bit only once when the Host Controller requests a data transfer to the Status Change endpoint. System software is responsible for determining state change history in such a case. Software sets this bit to 0 by writing a 1 to it.<br>1 = Change in Current Connect Status.<br>0 = No change. |
| 0   | <b>Current Connect Status</b> — RO. This value reflects the current state of the port, and may not correspond directly to the event that caused the Connect Status Change bit (Bit 1) to be set.<br>1 = Device is present on port.<br>0 = No device is present.   |



# SMBus Controller Registers (D31:F3)

# 11

## 11.1 PCI Configuration Registers (SMBUS—D31:F3)

Table 11-1. PCI Configuration Map (SMBUS—D31:F3)

| Offset | Mnemonic | Register Name/Function      | Default   | Type |
|--------|----------|-----------------------------|---|------|
| 00–01h | VID      | Vendor ID                   | 8086h   | RO   |
| 02–03h | DID      | Device ID                   | 2413h<br>(ICH: 82801AA)<br>2423h<br>(ICH0: 82801AB) | RO   |
| 04–05h | CMD      | Command Register            | 0000h   | R/W  |
| 06–07h | STA      | Device Status               | 0280h   | R/W  |
| 08h    | RID      | Revision ID                 | Note 1  | RO   |
| 09h    | PI       | Programming Interface       | 00h   | RO   |
| 0Ah    | SCC      | Sub Class Code              | 05h   | RO   |
| 0Bh    | BCC      | Base Class Code             | 0Ch   | RO   |
| 0Eh    | HEADTYP  | Header Type                 | 80h   | RO   |
| 20–23h | SMB_BASE | SMBus Base Address Register | 00000001h   | R/W  |
| 3Ch    | INTR_LN  | Interrupt Line              | 00h   | R/W  |
| 3Dh    | INTR_PN  | Interrupt Pin               | 02h   | RO   |
| 40h    | HOSTC    | Host Configuration          | 00h   | R/W  |

**NOTES:**

1. Refer to ICH Specification Updates for the value of the Revision ID.
2. Addresses that are not listed in the above table are reserved and read only.



### 11.1.4 STA—Device Status Register (SMBUS—D31:F3)

Address: 06–07h Attributes: R/W  
 Default Value: 0280h Size: 16 bits

| Bit  | Description  |
|------|--|
| 15   | Detected Parity Error (DPE) —RO: Reserved as '0'.  |
| 14   | Signaled System Error (SSE) —RO: Reserved as '0'.  |
| 13   | Received Master Abort (RMA) —RO: Reserved as '0'.  |
| 12   | Received Target Abort (RTA) —RO: Reserved as '0'.  |
| 11   | <b>Signaled Target-Abort Status—R/WC.</b> This bit is set when the function is targeted with a transaction that the ICH terminates with a target abort. Software resets STA to 0 by writing a 1 to this bit location..                     |
| 10:9 | <b>DEVSEL# Timing Status (DEVT)—RO.</b> This 2-bit field defines the timing for DEVSEL# assertion. These read only bits indicate the ICH's DEVSEL# timing when performing a positive decode. Note: ICH generates DEVSEL# with medium time. |
| 8    | Data Parity Error Detected—RO: Reserved as '0'.  |
| 7    | Fast Back-to-Back Capable—RO: Reserved as '1'.   |
| 6    | User Definable Features (UDF)—RO: Reserved as '0'.   |
| 5    | 66 MHz Capable—RO: Reserved as '0'.  |
| 4:0  | Reserved.  |

### 11.1.5 RID—Revision ID Register (SMBUS—D31:F3)

Offset Address: 08h Attribute: RO  
 Default Value: See ICH Spec. Updates Size: 8 bits

| Bit | Description  |
|-----|--|
| 7:0 | <b>Revision Identification Number.</b> 8-bit value that indicates the revision number for the ICH SMBus function. Refer to ICH Specification Updates for this value. |

### 11.1.6 PI—Programming Interface (SMBUS—D31:F3)

Address Offset: 09h Attribute: RO  
 Default Value: 00h Size: 8 bits

| Bit | Description                         |
|-----|-------------------------------------|
| 7:0 | <b>Programming Interface Value.</b> |





## 11.2 SMBus I/O Registers

Table 11-2. SMB I/O Registers

| Offset | Mnemonic  | Register Name/Function | Default | Access |
|--------|-----------|------------------------|---------|--------|
| 00h    | HST_STA   | Host Status            | 00h     | R/W    |
| 02h    | HST_CNT   | Host Control           | 00h     | R/W    |
| 03h    | HST_CMD   | Host Command           | 00h     | R/W    |
| 04h    | XMIT_SLVA | Transmit Slave Address | 00h     | R/W    |
| 05h    | HST_D0    | Host Data 0            | 00h     | R/W    |
| 06h    | HST_D1    | Host Data 1            | 00h     | R/W    |
| 07h    | BLOCK_DB  | Block Data Byte        | 00h     | R/W    |





## 11.2.2 HST\_CNT—Host Control Register

Register Offset: 02h  
Default Value: 00h

Attribute: R/W  
Size: 8-bits

| Bit | Description   |
|-----|---|
| 7   | Reserved.   |
| 6   | <b>START—WO.</b> This write-only bit is used to initiate the command described in the SMB_CMD field. All registers should be setup prior to writing a '1' to this bit position. This bit always reads zero. The HOST_BUSY bit in the Host Status register (offset 00h) can be used to identify when the ICH has finished the command.   |
| 5   | <b>LAST_BYTE.</b> This bit is used for Block Read commands.<br>1 = Software sets this bit to indicate that the next byte will be the last byte to be received for the block. The ICH will send a NOT ACK (instead of an ACK) after receiving the last byte.   |
| 4:2 | <b>SMB_CMD.</b> The bit encoding below indicates which command the ICH is to perform. If enabled, the ICH will generate an interrupt or SMI# when the command has completed. If the value is for a non-supported or reserved command, the ICH will set the device error (DEV_ERR) status bit and generate an interrupt when the START bit is set. The ICH will perform no command, and will not operate until DEV_ERR is cleared.<br><b>000 = Quick:</b> The slave address and read/write value (bit 0) are stored in the transmit slave address register.<br><b>001 = Byte:</b> This command uses the transmit slave address and command registers. Bit 0 of the slave address register determines if this is a read or write command.<br><b>010 = Byte Data:</b> This command uses the transmit slave address, command, and DATA0 registers. Bit 0 of the slave address register determines if this is a read or write command. If it is a read, the DATA0 register will contain the read data.<br><b>011 = Word Data:</b> This command uses the transmit slave address, command, DATA0 and DATA1 registers. Bit 0 of the slave address register determines if this is a read or write command. If it is a read, after the command completes, the DATA0 and DATA1 registers will contain the read data.<br><b>100 = Process Call:</b> This command uses the transmit slave address, command, DATA0 and DATA1 registers. Bit 0 of the slave address register determines if this is a read or write command. After the command completes, the DATA0 and DATA1 registers will contain the read data.<br><b>101 = Block:</b> This command uses the transmit slave address, command, DATA0 registers, and the Block Data Byte register. For block write, the count is stored in the DATA0 register and indicates how many bytes of data will be transferred. For block reads, the count is received and stored in the DATA0 register. Bit 0 of the slave address register selects if this is a read or write command. For writes, data is retrieved from the first n (where n is equal to the specified count) addresses of the SRAM array. For reads, the data is stored in the Block Data Byte register.<br><b>110 = I<sup>2</sup>C Read:</b> This command uses the transmit slave address, command, DATA0, DATA1 registers, and the Block Data Byte register. The read data is stored in the Block Data Byte register. The ICH will continue reading data until the NAK is received.<br><b>111 = Reserved</b> |
| 1   | <b>KILL.</b><br>1 = When set, kills the current host transaction taking place, sets the FAILED status bit, and asserts the interrupt (or SMI#) selected by the SMB_INTRSEL field.<br>0 = This bit, once set, must be cleared to allow the SMBus Host Controller to function normally.   |
| 0   | <b>INTREN.</b> Enable the generation of an interrupt or SMI# upon the completion of the command.  |



## 11.2.7 BLOCK\_DB—Block Data Byte Register

|                  |     |            |        |
|------------------|-----|------------|--------|
| Register Offset: | 07h | Attribute: | R/W    |
| Default Value:   | 00h | Size:      | 8 bits |

| Bit | Description  |
|-----|--|
| 7:0 | <p><b>Block Data Byte.</b> For Block Writes, the software will write the first byte to this register as part of the setup for this command. After the ICH has sent the Address, Command, and Byte Count fields, it will send the byte in the Block Data Byte register. After the byte has been sent, the ICH will set the BYTE_DONE_STS bit in the Host Status register. If there are more bytes to send, the software will then write in the next byte to the Block Data Byte register and the software will also clear the BYTE_DONE_STS bit. The ICH will then send the next byte. During the time from when a byte has been transmitted to when the next byte has been loaded, the ICH will insert wait-states on the SMBus/I<sup>2</sup>C.</p> <p>A similar process will be used for Block Reads. After receiving the byte count (which goes in the DATA 0 register), the first “data byte” goes in the Block Data Byte register and the ICH will generate an SMI# or interrupt (depending on configuration). The interrupt or SMI# handler will then read the byte and clear the BYTE_DONE_STS bit. This will free room for the next byte. During the time from when a byte is read to when the BYTE_DONE_STS bit is cleared, the ICH will insert wait-states on the SMBus/I<sup>2</sup>C.</p> |

# AC'97 Audio Controller Registers (D31:F5)

12

## 12.1 AC'97 Audio PCI Configuration Space (D31:F5)

Table 12-1. PCI Configuration Map (Audio—D31:F5)

| Offset  | Mnemonic | Register                                | Default   | Access     |
|---------|----------|---|---|------------|
| 00h–01h | VID      | Vendor Identification                   | 8086h   | RO         |
| 02h–03h | DID      | Device Identification                   | 2415h<br>(ICH: 82801AA)<br>2425h<br>(ICH0: 82801AB) | RO         |
| 04h–05h | PCICMD   | PCI Command                             | 0000  | R/W        |
| 06h–07h | PCISTA   | PCI Device Status                       | 0280h   | R/WC       |
| 08h     | RID      | Revision Identification                 | Note 1  | RO         |
| 09h     | PI       | Programming Interface                   | 00  | RO         |
| 0Ah     | SCC      | Sub Class Code                          | 01h   | RO         |
| 0Bh     | BCC      | Base Class Code                         | 04h   | RO         |
| 0Eh     | HEDTPY   | Header Type                             | 00  | RO         |
| 10h–13h | NAMBAR   | Native Audio Mixer Base Address         | 00000001h   | R/W        |
| 14h–17h | NABMBAR  | Native Audio Bus Mastering Base Address | 00000001h   | R/W        |
| 18h–1Bh |          | Reserved                                | 00000001h   |            |
| 1Ch–2Bh |          | Reserved                                |   |            |
| 2Ch–2Dh | SVID     | Subsystem Vendor ID                     | 0000h   | Write-Once |
| 2Eh–2Fh | SID      | Subsystem ID                            | 0000h   | Write-Once |
| 30h–3Bh |          | Reserved                                |   |            |
| 3Ch     | INTR_LN  | Interrupt Line                          | 00h   | R/W        |
| 3Dh     | INTR_PN  | Interrupt Pin                           | 02h   | RO         |
| 3Eh–FFh |          | Reserved                                |   |            |

**NOTES:**

1. Refer to ICH Specification Updates for the value of the Revision ID.
2. Addresses that are not listed in the above table are reserved and read only.

**12.1.1 VID—Vendor Identification Register (Audio—D31:F5)**

|                |         |             |         |
|----------------|---------|-------------|---------|
| Offset:        | 01h–00h | Attribute:  | RO      |
| Default Value: | 8086h   | Size:       | 16 Bits |
| Lockable:      | No      | Power Well: | Core    |

| Bit  | Description  |
|------|--|
| 15:0 | <b>Vendor ID Value.</b> This is a 16 bit value assigned to Intel |

**12.1.2 DID—Device Identification Register (Audio—D31:F5)**

|                |   |             |         |
|----------------|---|-------------|---------|
| Offset:        | 03h–02h                                       | Attribute:  | RO      |
| Lockable:      | No  | Size:       | 16 Bits |
| Default Value: | 2415h (ICH: 82801AA)<br>2425h (ICH0: 82901AB) | Power Well: | Core    |

| Bit  | Description             |
|------|-------------------------|
| 15:0 | <b>Device ID Value.</b> |

**12.1.3 PCICMD—PCI Command Register (Audio—D31:F5)**

|                 |         |             |         |
|-----------------|---------|-------------|---------|
| Address Offset: | 05h–04h | Attribute:  | R/W     |
| Default Value:  | 0000h   | Size:       | 16 bits |
| Lockable:       | No      | Power Well: | Core    |

PCICMD is a 16-bit control register. Refer to the PCI 2.1 specification for complete details on each bit.

| Bit   | Description   |
|-------|---|
| 15:10 | Reserved. <i>Read 0.</i>  |
| 9     | Fast Back to Back Enable (FBE). Not implemented. Hardwired to "0".  |
| 8     | SERR# Enable (SEN). Not implemented. Hardwired to "0".  |
| 7     | Wait Cycle Control (WCC). Not implemented. Hardwired to "0".  |
| 6     | Parity Error Response (PER). Not implemented. Hardwired to "0".   |
| 5     | VGA Palette Snoop (VPS). Not implemented. Hardwired to "0".   |
| 4     | Memory Write and Invalidate Enable (MWI). Not implemented. Hardwired to "0".  |
| 3     | Special Cycle Enable (SCE). Not implemented. Hardwired to "0".  |
| 2     | <b>Bus Master Enable (BME)—R/W.</b> Controls standard PCI bus mastering capabilities.<br>1 = Enable<br>0 = Disable.   |
| 1     | Memory Space (MS). Hardwired to "0", AC '97 does not respond to memory accesses   |
| 0     | <b>IOS (I/O Space).</b> This bit controls access to the I/O space registers.<br>1 = Enable access to I/O space. The Native PCI Mode Base Address register should be programmed prior to setting this bit.<br>0 = Disable (Default). |

### 12.1.4 PCISTA—PCI Device Status Register (Audio—D31:F5)

|               |         |             |         |
|---------------|---------|-------------|---------|
| Offset:       | 07h–06h | Attribute:  | R/WC    |
| Default Value | 0280h   | Size:       | 16 bits |
| Lockable:     | No      | Power Well: | Core    |

PCISTA is a 16-bit status register. Refer to the PCI 2.1 specification for complete details on each bit.

| Bit  | Description   |
|------|---|
| 15   | Detected Parity Error (DPE). Not implemented. Hardwired to "0".   |
| 14   | SERR# Status (SERRS). Not implemented. Hardwired to "0".  |
| 13   | <b>Master-Abort Status (MAS).</b><br>1 = Bus Master AC '97 2.1 interface function, as a master, generates a master abort.<br>0 = Software clears this bit by writing a "1" to this bit. |
| 12   | Reserved. Will always read as 0.  |
| 11   | <b>Signaled Target-Abort Status (STA).</b> Not implemented. Hardwired to "0".   |
| 10:9 | <b>DEVSEL# Timing Status (DEVT)—RO.</b> This 2-bit field reflects the ICH's DEVSEL# timing parameter. These bits indicate the ICH's DEVSEL# timing when performing a positive decode.   |
| 8    | Data Parity Detected (DPD). Not implemented. Hardwired to "0".  |
| 7    | Fast Back to back Capable (FBC). Hardwired to "1". This bit indicates that the ICH as a target is capable of fast back-to-back transactions.  |
| 6    | UDF Supported. Not implemented. Hardwired to "0".   |
| 5    | 66 MHz Capable. Hardwired to "0".   |
| 4:0  | Reserved. Read as 0's.  |

### 12.1.5 RID—Revision Identification Register (Audio—D31:F5)

|                |                       |             |        |
|----------------|-----------------------|-------------|--------|
| Offset:        | 08h                   | Attribute:  | RO     |
| Default Value: | See ICH Spec. Updates | Size:       | 8 Bits |
| Lockable:      | No                    | Power Well: | Core   |

| Bit | Description   |
|-----|---|
| 7:0 | <b>Revision ID Value.</b> 8-bit value that indicates the revision number for the AC'97 Audio Controller. Refer to ICH Specification Updates for this value. |

### 12.1.6 PI—Programming Interface Register (Audio—D31:F5)

|                |     |             |        |
|----------------|-----|-------------|--------|
| Offset:        | 09h | Attribute:  | RO     |
| Default Value: | 00h | Size:       | 8 bits |
| Lockable:      | No  | Power Well: | Core   |

| Bit | Description                   |
|-----|-------------------------------|
| 7:0 | <b>Programming Interface.</b> |

**12.1.7 SCC—Sub Class Code Register (Audio—D31:F5)**

|                 |     |             |        |
|-----------------|-----|-------------|--------|
| Address Offset: | 0Ah | Attribute:  | RO     |
| Default Value:  | 01h | Size:       | 8 bits |
| Lockable:       | No  | Power Well: | Core   |

This register indicates that the device is an audio device, in the context of a multimedia device (Base Class Code = 04h).

| Bit | Description                               |
|-----|---|
| 7:0 | <b>Sub Class Code.</b> 01h = Audio Device |

**12.1.8 BCC—Base Class Code Register (Audio—D31:F5)**

|                 |     |             |        |
|-----------------|-----|-------------|--------|
| Address Offset: | 0Bh | Attribute:  | RO     |
| Default Value:  | 04h | Size:       | 8 bits |
| Lockable:       | No  | Power Well: | Core   |

| Bit | Description                                     |
|-----|---|
| 7:0 | <b>Base Class Code.</b> 04h = Multimedia device |

**12.1.9 HEADTYP—Header Type Register (Audio—D31:F5)**

|                 |     |             |        |
|-----------------|-----|-------------|--------|
| Address Offset: | 0Eh | Attribute:  | RO     |
| Default Value:  | 00h | Size:       | 8 bits |
| Lockable:       | No  | Power Well: | Core   |

| Bit | Description  |
|-----|--|
| 7:0 | <b>Header Type Value.</b> This value is always read as zero. |



### 12.1.10 NAMBAR—Native Audio Mixer Base Address Register (Audio—D31:F5)

|                 |           |             |         |
|-----------------|-----------|-------------|---------|
| Address Offset: | 10h–13h   | Attribute:  | R/W     |
| Default Value:  | 00000001h | Size:       | 32 bits |
| Lockable:       | No        | Power Well: | Core    |

The Native PCI Mode Audio function uses PCI Base Address register #1 to request a contiguous block of I/O space that is to be used for the Native Audio Mixer software interface. The mixer requires 256 bytes of I/O space. Native Audio Mixer and Modem codec I/O registers are located from 00h to 7Fh and reside in the codec. Access to these registers will be decoded by the AC '97 controller and forwarded over the AC-link to the codec. The codec will then respond with the register value.

In the case of the split codec implementation accesses to the different codecs are differentiated by the controller by using address offsets 00h–7Fh for the primary codec and address offsets 80h–FEh for the secondary codec.

For description of these I/O registers, refer to the AC'97 specification.

| Bit   | Description  |
|-------|--|
| 31:16 | Hardwired to 0's   |
| 15:8  | <b>Base Address.</b> These bits are used in the I/O space decode of the Native Audio Mixer interface registers. The number of upper bits that a device actually implements depends on how much of the address space the device will respond to. For the AC '97 mixer, the upper 16 bits are hardwired to 0, while bits 15:8 are programmable. This configuration yields a maximum I/O block size of 256 bytes for this base address. |
| 7:1   | Reserved. Read as 0's.   |
| 0     | <b>Resource Type Indicator (RTE).</b> This bit is set to one, indicating a request for I/O space.  |

### 12.1.11 NABMBAR—Native Audio Bus Mastering Base Address Register (Audio—D31:F5)

|                 |           |             |         |
|-----------------|-----------|-------------|---------|
| Address Offset: | 14h–17h   | Attribute:  | R/W     |
| Default Value:  | 00000001h | Size:       | 32 bits |
| Lockable:       | No        | Power Well: | Core    |

The Native PCI Mode Audio function uses PCI Base Address register #1 to request a contiguous block of I/O space that is to be used for the Native Mode Audio software interface.

| Bit   | Description   |
|-------|---|
| 31:16 | Hardwired to 0's  |
| 15:6  | <b>Base Address.</b> These bits are used in the I/O space decode of the Native Audio Bus Mastering interface registers. The number of upper bits that a device actually implements depends on how much of the address space the device will respond to. For AC '97 bus mastering, the upper 16 bits are hardwired to 0, while bits 15:6 are programmable. This configuration yields a maximum I/O block size of 64 bytes for this base address. |
| 5:1   | Reserved. Read as 0's.  |
| 0     | <b>Resource Type Indicator (RTE).</b> This bit is set to one, indicating a request for I/O space.   |

**12.1.12 SVID—Subsystem Vendor ID Register (Audio—D31:F5)**

|                 |         |             |            |
|-----------------|---------|-------------|------------|
| Address Offset: | 2Dh–2Ch | Attribute:  | Write-Once |
| Default Value:  | 0000h   | Size:       | 16 bits    |
| Lockable:       | No      | Power Well: | Core       |

This register should be implemented for any function that could be instantiated more than once in a given system, for example, a system with 2 audio subsystems, one down on the motherboard and the other plugged into a PCI expansion slot, should have the SVID register implemented. The SVID register, in combination with the Subsystem ID register, enable the operating environment to distinguish one audio subsystem from the other(s).

This register is implemented as write-once register. Once a value is written to it, the value can be read back. Any subsequent writes will have no effect.

| Bit  | Description                       |
|------|-----------------------------------|
| 15:0 | <b>Subsystem Vendor ID Value.</b> |

**12.1.13 SID—Subsystem ID Register (Audio—D31:F5)**

|                 |         |             |         |
|-----------------|---------|-------------|---------|
| Address Offset: | 2Fh–2Eh | Attribute:  | RO      |
| Default Value:  | 0000h   | Size:       | 16 bits |
| Lockable:       | No      | Power Well: | Core    |

This register should be implemented for any function that could be instantiated more than once in a given system, for example, a system with 2 audio subsystems, one down on the motherboard and the other plugged into a PCI expansion slot. The SID register, in combination with the Subsystem Vendor ID register make it possible for the operating environment to distinguish one audio subsystem from the other(s).

This register is implemented as write-once register. Once a value is written to it, the value can be read back. Any subsequent writes will have no effect.

| Bit  | Description                |
|------|----------------------------|
| 15:0 | <b>Subsystem ID Value.</b> |

**12.1.14 INTR\_LN—Interrupt Line Register (Audio—D31:F5)**

|                 |     |             |        |
|-----------------|-----|-------------|--------|
| Address Offset: | 3Ch | Attribute:  | R/W    |
| Default Value:  | 00h | Size:       | 8 bits |
| Lockable:       | No  | Power Well: | Core   |

This register indicates which PCI interrupt line is used for the AC'97 module interrupt.

| Bit | Description   |
|-----|---|
| 7:0 | <b>Interrupt Line:</b> This data is not used by the ICH. It is used to communicate to software the interrupt line that the interrupt pin is connected to. |

### 12.1.15 INTR\_PN—Interrupt Pin Register (Audio—D31:F5)

|                 |     |             |        |
|-----------------|-----|-------------|--------|
| Address Offset: | 3Dh | Attribute:  | RO     |
| Default Value:  | 02h | Size:       | 8 bits |
| Lockable:       | No  | Power Well: | Core   |

This register indicates which PCI interrupt pin is used for the AC '97 module interrupt. The AC '97 interrupt is internally OR'ed to the interrupt controller with the PIRQB# signal.

| Bit | Description  |
|-----|--|
| 7:3 | Reserved.  |
| 2:0 | <b>AC '97 Interrupt Routing.</b> Hardwired to 010b to select PIRQB#. |

## 12.2 AC'97 Audio I/O Space (D31:F5)

The AC'97 I/O space includes Native Audio Bus Master Registers and Native Mixer Registers. Table 12-2 shows the register addresses for the audio mixer registers.

**Table 12-2. ICH Audio Mixer Register Configuration**

| Register | NAMBAR Exposed Registers (D31:F5) |
|----------|-----------------------------------|
| Pri.     | Name                              |
| 00h      | Reset                             |
| 02h      | Master Volume Mute                |
| 04h      | Headphone Volume Mute             |
| 06h      | Master Volume Mono Mute           |
| 08h      | Master Tone (R & L)               |
| 0Ah      | PC_BEEP Volume Mute               |
| 0Ch      | Phone Volume Mute                 |
| 0Eh      | Mic Volume Mute                   |
| 10h      | Line In Volume Mute               |
| 12h      | CD Volume Mute                    |
| 14h      | Video Volume Mute                 |
| 16h      | Aux Volume Mute                   |
| 18h      | PCM Out Volume Mute               |
| 1Ah      | Record Select                     |
| 1Ch      | Record Gain Mute                  |
| 1Eh      | Record Gain Mic Mute              |
| 20h      | General Purpose                   |
| 22h      | 3D Control                        |
| 24h      | AC'97 RESERVED                    |
| 26h      | Powerdown Ctrl/Stat               |
| 28h      | Extended Audio                    |
| 2Ah      | Extended Audio Ctrl/Stat          |

Table 12-2. ICH Audio Mixer Register Configuration

| Register   | NAMBAR Exposed Registers (D31:F5)  |
|------------|------------------------------------|
| Pri.       | Name                               |
| 2Ch        | PCM Front DAC Rate                 |
| <i>2Eh</i> | <i>PCM Surround DAC Rate</i>       |
| <i>30h</i> | <i>PCM LFE DAC Rate</i>            |
| 32h        | PCM LR ADC Rate                    |
| 34h        | MIC ADC Rate                       |
| <i>36h</i> | <i>6Ch Vol: C, LFE Mute</i>        |
| <i>38h</i> | <i>6Ch Vol: L, R Surround Mute</i> |
| 3Ah:56h    | Intel RESERVED                     |
| <b>58h</b> | <b>Vendor Reserved</b>             |
| <b>7Ah</b> | <b>Vendor Reserved</b>             |
| <b>7Ch</b> | <b>Vendor ID1</b>                  |
| <b>7Eh</b> | <b>Vendor ID2</b>                  |

**NOTE:**

1. Registers in bold are multiplexed between audio and modem functions
2. Registers in italics are for functions not supported by the ICH. The ICH does not support audio as a secondary codec.
3. Software should not try to access reserved registers

The Bus Master registers are located from offset + 00h to offset + 51h and reside in the AC '97 controller. Accesses to these registers do NOT cause the cycle to be forwarded over the AC-link to the codec.

In the case of the split codec implementation accesses to the different codecs are differentiated by the controller by using address offsets 00h–7Fh for the primary codec and address offsets 80h–FEh for the secondary codec.

The Global Control (GLOB\_CNT) and Global Status (GLOB\_STA) registers are aliased to the same global registers in the audio and modem I/O space. Therefore a read/write to these registers in either audio or modem I/O space affects the same physical register.

Bus Mastering registers exist in I/O space and reside in the AC '97 controller. The three channels, PCM in, PCM out, and Mic in, each have their own set of Bus Mastering registers. The following register descriptions apply to all three channels. The register definition section titles use a generic "x\_" in front of the register to indicate that the register applies to all three channels. The naming prefix convention used in Table 12-3 and in the register description I/O address is as follows:

PI = PCM in channel  
 PO = PCM out channel  
 MC = Mic in channel.

**Table 12-3. Native Audio Bus Master Control Registers**

| Offset | Mnemonic | Name   | Default   | Access |
|--------|----------|--|-----------|--------|
| 00h    | PI_BDBAR | PCM In Buffer Descriptor list Base Address Register  | 00000000h | R/W    |
| 04h    | PI_CIV   | PCM In Current Index Value                           | 00h       | RO     |
| 05h    | PI_LVI   | PCM In Last Valid Index                              | 00h       | R/W    |
| 06h    | PI_SR    | PCM In Status Register                               | 0001h     | R/W    |
| 08h    | PI_PICB  | PCM In Position In Current Buffer                    | 0000h     | RO     |
| 0Ah    | PI_PIV   | PCM In Prefetched Index Value                        | 00h       | RO     |
| 0Bh    | PI_CR    | PCM In Control Register                              | 00h       | R/W    |
| 10h    | PO_BDBAR | PCM Out Buffer Descriptor list Base Address Register | 00000000h | R/W    |
| 14h    | PO_CIV   | PCM Out Current Index Value                          | 00h       | RO     |
| 15h    | PO_LVI   | PCM Out Last Valid Index                             | 00h       | R/W    |
| 16h    | PO_SR    | PCM Out Status Register                              | 0001h     | R/W    |
| 18h    | PO_PICB  | PCM Out Position In Current Buffer                   | 0000h     | RO     |
| 1Ah    | PO_PIV   | PCM Out Prefetched Index Value                       | 00h       | RO     |
| 1Bh    | PO_CR    | PCM Out Control Register                             | 00h       | R/W    |
| 20h    | MC_BDBAR | Mic. In Buffer Descriptor list Base Address Register | 00000000h | R/W    |
| 24h    | PM_CIV   | Mic. In Current Index Value                          | 00h       | RO     |
| 25h    | MC_LVI   | Mic. In Last Valid Index                             | 00h       | R/W    |
| 26h    | MC_SR    | Mic. In Status Register                              | 0001h     | R/W    |
| 28h    | MC_PICB  | Mic In Position In Current Buffer                    | 0000h     | RO     |
| 2Ah    | MC_PIV   | Mic. In Prefetched Index Value                       | 00h       | RO     |
| 2Bh    | MC_CR    | Mic. In Control Register                             | 00h       | R/W    |
| 2Ch    | GLOB_CNT | Global Control                                       | 00000000h | R/W    |
| 30h    | GLOB_STA | Global Status  | 00000000h | RO     |
| 34h    | ACC_SEMA | Codec Write Semaphore Register                       | 00h       | R/W    |

### 12.2.1 x\_BDBAR—Buffer Descriptor Base Address Register

I/O Address: NABMBAR + 00h (PIBDBAR), Attribute: R/W  
 NABMBAR + 10h (POBDBAR),  
 NABMBAR + 20h (MCBDBAR)  
 Default Value: 00000000h Size: 32 bits  
 Lockable: No Power Well: Core

| Bit  | Description   |
|------|---|
| 31:3 | <b>Buffer Descriptor Base address[31:3]</b> . These bits represent address bits 31:3. The data should be aligned on 8 byte boundaries. Each buffer descriptor is 8 bytes long and the list can contain a maximum of 32 entries. |
| 2:0  | Hardwired to 0.   |

### 12.2.2 x\_CIV—Current Index Value Register

I/O Address: NABMBAR + 04h (PICIV), Attribute: RO  
 NABMBAR + 14h (POCIV),  
 NABMBAR + 24h (MCCIV)  
 Default Value: 00h Size: 8 bits  
 Lockable: No Power Well: Core

Software can read the registers at offsets 04h, 05h and 06h simultaneously by performing a single 32 bit read from address offset 04h. Software can also read this register individually by doing a single 8 bit read to offset 04h.

| Bit | Description  |
|-----|--|
| 7:5 | Hardwired to 0   |
| 4:0 | <b>Current Index Value[4:0]</b> . These bits represent which buffer descriptor within the list of 32 descriptors is currently being processed. As each descriptor is processed, this value is incremented. The value rolls over after it reaches 31. |

### 12.2.3 x\_LVI—Last Valid Index Register

I/O Address: NABMBAR + 05h (PILVI), Attribute: R/W  
 NABMBAR + 15h (POLVI),  
 NABMBAR + 25h (MCLVI)  
 Default Value: 00h Size: 8 bits  
 Lockable: No Power Well: Core

Software can read the registers at offsets 04h, 05h and 06h simultaneously by performing a single 32 bit read from address offset 04h. Software can also read this register individually by doing a single 8 bit read to offset 05h.

| Bit | Description   |
|-----|---|
| 7:5 | Hardwired to 0.   |
| 4:0 | <b>Last Valid Index[4:0]</b> . This value represents the last valid descriptor in the list. This value is updated by the software each time it prepares a new buffer and adds it to the list. |

## 12.2.4 x\_SR—Status Register

|                |  |             |         |
|----------------|--|-------------|---------|
| I/O Address:   | NABMBAR + 06h (PISR),<br>NABMBAR + 16h (POSR),<br>NABMBAR + 26h (MCSR) | Attribute:  | R/W     |
| Default Value: | 0001h  | Size:       | 16 bits |
| Lockable:      | No   | Power Well: | Core    |

Software can read these 3 registers simultaneously by scheduling a single 32 bit read from address offset 04h. Software can also read this individual register by performing a 16 bit read from 06h.

| Bit  | Description   |
|------|---|
| 15:5 | Reserved.   |
| 4    | <p><b>FIFO error (FIFOE).</b> The ICH will set the FIFOE bit if the under-run or overrun occurs when there are more valid buffers to process.</p> <p>1 = FIFO error occurs.<br/>0 = Cleared by writing a "1" to this bit position.</p> <p><b>NOTES:</b></p> <ol style="list-style-type: none"> <li><b>PISR Register:</b> FIFO error indicates a FIFO overrun. The FIFO pointers don't increment, the incoming data is not written into the FIFO, thus is lost.</li> <li><b>POSR Register:</b> FIFO error indicates a FIFO under-run. The sample transmitted in this case should be the last valid sample.</li> </ol>  |
| 3    | <p><b>Buffer Completion Interrupt Status (BCIS).</b></p> <p>1 = Set by the hardware after the last sample of a buffer has been processed, AND if the Interrupt on Completion (IOC) bit is set in the command byte of the buffer descriptor. It remains active until cleared by software.<br/>0 = Cleared by writing a "1" to this bit position.</p>   |
| 2    | <p><b>Last Valid Buffer Completion Interrupt (LVBCI).</b></p> <p>1 = Last valid buffer has been processed. It remains active until cleared by software. This bit indicates the occurrence of the event signified by the last valid buffer being processed. Thus this is an event status bit that can be cleared by software once this event has been recognized. This event will cause an interrupt if the enable bit in the Control Register is set. The interrupt is cleared when the software clears this bit.</p> <p>In the case of <i>Transmits</i> (PCM out, Modem out) this bit is set, after the last valid buffer has been fetched (not after transmitting it). While in the case of <i>Receives</i>, this bit is set after the data for the last buffer has been written to memory.</p> <p>0 = Cleared by writing a "1" to this bit position.</p> |
| 1    | <p><b>Current Equals Last Valid (CELV).</b></p> <p>1 = Current Index is equal to the value in the Last Valid Index Register, and the buffer pointed to by the CIV has been processed (i.e., after the last valid buffer has been processed). This bit is very similar to bit 2, except this bit reflects the state rather than the event. This bit reflects the state of the controller, and remains set until the controller exits this state.</p> <p>0 = Cleared by hardware when controller exists state (i.e., until a new value is written to the LVI register.)</p>   |
| 0    | <p><b>DMA Controller Halted (DCH).</b></p> <p>1 = Halted. This could happen because of the Start/Stop bit being cleared, or it could happen once the controller has processed the last valid buffer (in which case it will set bit 1 and halt).</p>   |

### 12.2.5 x\_PICB—Position In Current Buffer Register

I/O Address: NABMBAR + 08h (PIPICB), Attribute: RO  
 NABMBAR + 18h (POPICB),  
 NABMBAR + 28h (MCPICB)  
 Default Value: 0000h Size: 16 bits  
 Lockable: No Power Well: Core

| Bit  | Description   |
|------|---|
| 15:0 | <b>Position In Current Buffer[15:0]</b> . These bits represent the number of dwords left to be processed in the current buffer. Once again, this means, the number of samples not yet read from memory (in the case of reads from memory) or not yet written to memory (in the case of writes to memory), irrespective of the number of samples that have been transmitted/received across AC-link. |

### 12.2.6 x\_PIV—Prefetched Index Value Register

I/O Address: NABMBAR + 0Ah (PIPIV), Attribute: RO  
 NABMBAR + 1Ah (POPIV),  
 NABMBAR + 2Ah (MCPIV)  
 Default Value: 00h Size: 8 bits  
 Lockable: No Power Well: Core

| Bit | Description  |
|-----|--|
| 7:5 | Hardwired to 0.  |
| 4:0 | <b>Prefetched Index Value[4:0]</b> . These bits represent which buffer descriptor in the list has been prefetched. The bits in this register are also modulo 32 and roll over after they reach 31. |



## 12.2.7 x\_CR—Control Register

|                |  |             |        |
|----------------|--|-------------|--------|
| I/O Address:   | NABMBAR + 0Bh (PICR),<br>NABMBAR + 1Bh (POCR),<br>NABMBAR + 2Bh (MCCR) | Attribute:  | R/W    |
| Default Value: | 00h  | Size:       | 8 bits |
| Lockable:      | No   | Power Well: | Core   |

| Bit | Description  |
|-----|--|
| 7:5 | Reserved.  |
| 4   | <b>Interrupt On Completion Enable (IOCE).</b> This bit controls whether or not an interrupt occurs when a buffer completes with the IOC bit set in its descriptor.   |
| 3   | <b>FIFO Error Interrupt Enable (FEIE).</b> This bit controls whether the occurrence of a FIFO error will cause an interrupt or not.<br>1 = Enable. Interrupt will occur<br>0 = Disable. Bit 4 in the Status Register will be set, but the interrupt will not occur.  |
| 2   | <b>Last Valid Buffer Interrupt Enable (LVBIE).</b> This bit controls whether the completion of the last valid buffer will cause an interrupt or not.<br>1 = Enable.<br>0 = Disable. Bit 2 in the Status register will still be set, but the interrupt will not occur.  |
| 1   | <b>Reset Registers (RR).</b><br>1 = Contents of all Bus master related registers to be reset, except the interrupt enable bits (bit 4,3,2 of this register). Software needs to set this bit but need not clear it since the bit is self clearing. This bit must be set only when the Run/Pause bit is cleared. Setting it when the Run bit is set will cause undefined consequences.<br>0 = Removes reset condition. |
| 0   | <b>Run/Pause Bus master (RPBM).</b><br>1 = Run. Bus master operation starts.<br>0 = Pause bus master operation. This results in all state information being retained (i.e., master mode operation can be stopped and then resumed).  |

## 12.2.8 GLOB\_CNT—Global Control Register

I/O Address: NABMBAR + 2Ch      Attribute: R/W  
 Default Value: 00000000h      Size: 32 bits  
 Lockable: No      Power Well: Core

| Bit  | Description   |
|------|---|
| 31:6 | Reserved.   |
| 5    | <b>Secondary Resume Interrupt Enable</b><br>1 = Enable an interrupt to occur when the secondary codec causes a resume event on the AC-link.<br>0 = Disable  |
| 4    | <b>Primary Resume Interrupt Enable</b><br>1 = Enable an interrupt to occur when the primary codec causes a resume event on the AC-link.<br>0 = Disable  |
| 3    | <b>ACLINK Shut Off</b><br>1 = Drive all AC'97 outputs low and turn off all AC'97 input buffer enables   |
| 2    | <b>AC'97 Warm Reset.</b><br>1 = Writing a "1" to this bit causes a warm reset to occur on the AC-link. The warm reset will awaken a suspended codec without clearing its internal registers. If software attempts to perform a warm reset while bit_clk is running, the write will be ignored and the bit will not change. This bit is self-clearing (it remains set until the reset completes and bit_clk is seen on the ACLink, after which it clears itself).  |
| 1    | <b>AC '97 Cold Reset#.</b><br>0 = Writing a "0" to this bit causes a cold reset to occur throughout the AC '97 circuitry. All data in the controller and the codec will be lost. Software needs to clear this bit no sooner than the minimum number of ms have elapsed. This bit defaults to 0 and hence after reset, the driver needs to set this bit to a 1. The value of this bit is retained after suspends; hence, if this bit is set to a 1 prior to suspending, a cold reset is not generated automatically upon resuming.<br><br><b>NOTE:</b> This bit is in the Resume well, not in the Core well. |
| 0    | <b>GPI Interrupt Enable (GIE).</b> This bit controls whether the change in status of any GPI causes an interrupt.<br>1 = The change on value of a GPI causes an interrupt and sets bit 0 of the Global Status Register.<br>0 = Bit 0 of the Global Status Register is set, but an interrupt is not generated.   |

## 12.2.9 GLOB\_STA—Global Status Register

I/O Address: NABMBAR + 30h      Attribute: R/W  
 Default Value: 00000000h      Size: 32 bits  
 Lockable: No      Power Well: Core

| Bit   | Description   |
|-------|---|
| 31:18 | Reserved.   |
| 17    | <b>MD3:</b> Power down semaphore for Modem. This bit is used by software in conjunction with the AD3 bit to coordinate the entry of the two codecs into D3 state. This bit resides in the resume well and maintain context across power states. |
| 16    | <b>AD3:</b> Power down semaphore for Audio. This bit is used by software in conjunction with the MD3 bit to coordinate the entry of the two codecs into D3 state. This bit resides in the resume well and maintain context across power states. |

| Bit | Description   |
|-----|---|
| 15  | <b>Read Completion Status:</b> This bit indicates the status of codec read completions.<br>1 = A codec read results in a time-out. The bit remains set until being cleared by software.<br>0 = A codec read completes normally.   |
| 14  | <b>Bit 3 of slot 12:</b> Display bit 3 of the most recent slot 12. This bit is Read Only.   |
| 13  | <b>Bit 2 of slot 12:</b> Display bit 2 of the most recent slot 12. This bit is Read Only.   |
| 12  | <b>Bit 1 of slot 12:</b> Display bit 1 of the most recent slot 12. This bit is Read Only.   |
| 11  | <b>Secondary Resume Interrupt.</b> This bit indicates that a resume event occurred on AC_SDIN[1].<br>1 = Resume event occurred<br>0 = Cleared by writing a 1 to this bit position.  |
| 10  | <b>Primary Resume Interrupt.</b> This bit indicates that a resume event occurred on AC_SDIN[0].<br>1 = Resume event occurred<br>0 = Cleared by writing a 1 to this bit position.  |
| 9   | <b>Secondary Codec Ready (SCR).</b> Reflects the state of the codec ready bit in AC_SDIN[1]. Bus masters ignore the condition of the codec ready bits. Software must check this bit before starting the bus masters. Once the codec is "ready", it must never go "not ready" spontaneously.             |
| 8   | <b>Primary Codec Ready (PCR).</b> Reflects the state of the codec ready bit in AC_SDIN [0]. Bus masters ignore the condition of the codec ready bits. Software must check this bit before starting the bus masters. Once the codec is "ready", it must never go "not ready" spontaneously.              |
| 7   | <b>Mic In Interrupt (MINT).</b> This bit indicates that one of the Mic in channel interrupts occurred.<br>1 = Interrupt occurred.<br>0 = When the specific interrupt is cleared, this bit will be cleared.  |
| 6   | <b>PCM Out Interrupt (POINT).</b> This bit indicates that one of the PCM out channel interrupts occurred.<br>1 = Interrupt occurred.<br>0 = When the specific interrupt is cleared, this bit will be cleared.   |
| 5   | <b>PCM In Interrupt (PIINT).</b> This bit indicates that one of the PCM in channel interrupts occurred.<br>1 = Interrupt occurred.<br>0 = When the specific interrupt is cleared, this bit will be cleared.   |
| 4:3 | Reserved  |
| 2   | <b>Modem Out Interrupt (MOINT).</b> This bit indicates that one of the modem out channel interrupts occurred.<br>1 = Interrupt occurred.<br>0 = When the specific interrupt is cleared, this bit will be cleared.   |
| 1   | <b>Modem In Interrupt (MIINT).</b> This bit indicates that one of the modem in channel interrupts occurred.<br>1 = Interrupt occurred.<br>0 = When the specific interrupt is cleared, this bit will be cleared.   |
| 0   | <b>GPI Status Change Interrupt (GSCI).</b> This bit reflects the state of bit 0 in slot 12, and is set whenever bit 0 of slot 12 is set. This happens when the value of any of the GPIOs currently defined as inputs changes.<br>1 = Input changed.<br>0 = Cleared by writing a 1 to this bit position. |

### 12.2.10 CAS—Codec Access Semaphore Register

|                |               |             |        |
|----------------|---------------|-------------|--------|
| I/O Address:   | NABMBAR + 34h | Attribute:  | R/W    |
| Default Value: | 00h           | Size:       | 8 bits |
| Lockable:      | No            | Power Well: | Core   |

| Bit | Description  |
|-----|--|
| 7:1 | Reserved.  |
| 0   | <p><b>Codec Access Semaphore (CAS).</b> This bit is read by software to check whether a codec access is currently in progress.</p> <p>1 = The act of reading this register sets this bit to 1. The driver that read this bit can then perform an I/O access. Once the access is completed, hardware automatically clears this bit.</p> <p>0 = No access in progress.</p> |

# AC '97 Modem Controller Registers (D31:F6)

13

## 13.1 AC '97 Modem PCI Configuration Space (D31:F6)

Table 13-1. PCI Configuration Map (Modem—D31:F6)

| Offset  | Mnemonic | Register                 | Default   | Access     |
|---------|----------|--------------------------|---|------------|
| 00h–01h | VID      | Vendor Identification    | 8086  | RO         |
| 02h–03h | DID      | Device Identification    | 2416h<br>(ICH: 82801AA)<br>2426h<br>(ICH0: 82801AB) | RO         |
| 04h–05h | PCICMD   | PCI Command              | 0000  | R/W        |
| 06h–07h | PCISTA   | PCI Device Status        | 0280h   | R/WC       |
| 08h     | RID      | Revision Identification  | Note 1  | RO         |
| 09h     | PI       | Programming Interface    | 00  | RO         |
| 0Ah     | SCC      | Sub Class Code           | 03h   | RO         |
| 0Bh     | BCC      | Base Class Code          | 07h   | RO         |
| 0Eh     | HEDTYP   | Header Type              | 00  | RO         |
| 0Fh     | —        | Reserved                 | —   | —          |
| 10h–13h | MMBAR    | Modem Mixer Base Address | 00000001h   | R/W        |
| 14h–17h | MBAR     | Modem Base Address       | 00000001h   | R/W        |
| 18h–1Bh | —        | Reserved                 | 00000001h   | —          |
| 1Ch–2Bh | —        | Reserved                 | —   | —          |
| 2Ch–2Dh | SVID     | Subsystem Vendor ID      | 0000h   | Write-Once |
| 2Eh–2Fh | SID      | Subsystem ID             | 0000h   | Write-Once |
| 30h–3Bh | —        | Reserved                 | —   | —          |
| 3Ch     | INTR_LN  | Interrupt Line           | 00h   | RO         |
| 3Dh     | INT_PN   | Interrupt Pin            | 02h   | RO         |

**NOTES:**

1. Refer to ICH Specification Updates for the value of the Revision ID.
2. Addresses that are not listed in the above table are reserved and read only.

**13.1.1 VID—Vendor Identification Register (Modem—D31:F6)**

|                 |         |             |         |
|-----------------|---------|-------------|---------|
| Address Offset: | 01h–00h | Attribute:  | RO      |
| Default Value:  | 8086    | Size:       | 16 Bits |
| Lockable:       | No      | Power Well: | Core    |

| Bit  | Description             |
|------|-------------------------|
| 15:0 | <b>Vendor ID Value.</b> |

**13.1.2 DID—Device Identification Register (Modem—D31:F6)**

|                 |   |             |         |
|-----------------|---|-------------|---------|
| Address Offset: | 03h–02h                                       | Attribute:  | RO      |
| Default Value:  |   | Size:       | 16 Bits |
| Lockable:       | No  | Power Well: | Core    |
| Default Value:  | 2416h (ICH: 82801AA)<br>2426h (ICH0: 82901AB) |             |         |

| Bit  | Description             |
|------|-------------------------|
| 15:0 | <b>Device ID Value.</b> |

**13.1.3 PCICMD—PCI Command Register (Modem—D31:F6)**

|                 |         |             |         |
|-----------------|---------|-------------|---------|
| Address Offset: | 05h–04h | Attribute:  | R/W     |
| Default Value:  | 0000h   | Size:       | 16 bits |
| Lockable:       | No      | Power Well: | Core    |

PCICMD is a 16-bit control register. Refer to the PCI 2.1 specification for complete details on each bit.

| Bit   | Description  |
|-------|--|
| 15:10 | Reserved. Read 0.  |
| 9     | Fast Back to Back Enable (FBE). Not implemented. Hardwired to "0".   |
| 8     | SERR# Enable (SEN). Not implemented. Hardwired to "0".   |
| 7     | Wait Cycle Control (WCC). Not implemented. Hardwired to "0".   |
| 6     | Parity Error Response (PER). Not implemented. Hardwired to "0".  |
| 5     | VGA Palette Snoop (VPS). Not implemented. Hardwired to "0".  |
| 4     | Memory Write and Invalidate Enable (MWI). Not implemented. Hardwired to "0".   |
| 3     | Special Cycle Enable (SCE). Not implemented. Hardwired to "0".   |
| 2     | <b>Bus Master Enable (BME)—R/W.</b> Controls standard PCI bus mastering capabilities.<br>1 = Enable<br>0 = Disable.  |
| 1     | Memory Space (MS). Hardwired to "0", AC '97 does not respond to memory accesses.   |
| 0     | <b>I/O Space (IOS)—R/W.</b> This bit controls access to the I/O space registers.<br>1 = Enable access to I/O space. The Native PCI Mode Base Address register should be programmed prior to setting this bit.<br>0 = Disable access. (default = 0) |

### 13.1.4 PCISTA—Device Status Register (Modem—D31:F6)

|                 |         |             |         |
|-----------------|---------|-------------|---------|
| Address Offset: | 07h–06h | Attribute:  | R/WC    |
| Default Value:  | 0280h   | Size:       | 16 bits |
| Lockable:       | No      | Power Well: | Core    |

PCISTA is a 16-bit status register. Refer to the PCI 2.1 specification for complete details on each bit.

| Bit  | Description  |
|------|--|
| 15   | DPE (Detected Parity Error)—RO. Not implemented. Hardwired to "0".   |
| 14   | SERRS (SERR# Status) —RO. Not implemented. Hardwired to "0".   |
| 13   | <b>MAS (Master-Abort Status)—R/WC.</b><br>1 = Bus Master AC '97 interface function, as a master, generates a master abort.<br>0 = Software clears this bit by writing a "1" to this bit. |
| 12   | Reserved. Read as "0".   |
| 11   | STA (Signaled Target-Abort Status) —RO. Not implemented. Hardwired to "0".   |
| 10:9 | DEVT (DEVSEL# Timing Status)—RO. This 2-bit field reflects the ICH's DEVSEL# timing parameter. These read only bits indicate the ICH's DEVSEL# timing when performing a positive decode. |
| 8    | DPD (Data Parity Detected) —RO. Not implemented. Hardwired to "0".   |
| 7    | FBC (Fast Back to back Capable) —RO. Hardwired to "1". This bit indicates that the ICH as a target is capable of fast back-to-back transactions.   |
| 6    | UDF Supported—RO. Not implemented. Hardwired to "0".   |
| 5    | 66 MHz Capable—RO. Hardwired to "0".   |
| 4:0  | Reserved. Read as 0's.   |

### 13.1.5 RID—Revision Identification Register (Modem—D31:F6)

|                 |                       |             |        |
|-----------------|-----------------------|-------------|--------|
| Address Offset: | 08h                   | Attribute:  | RO     |
| Default Value:  | See ICH Spec. Updates | Size:       | 8 Bits |
| Lockable:       | No                    | Power Well: | Core   |

| Bit | Description   |
|-----|---|
| 7:0 | <b>Revision ID Value.</b> 8-bit value that indicates the revision number for the AC'97 Modem Controller. Refer to ICH Specification Updates for this value. |

### 13.1.6 PI—Programming Interface Register (Modem—D31:F6)

|                 |     |             |        |
|-----------------|-----|-------------|--------|
| Address Offset: | 09h | Attribute:  | RO     |
| Default Value:  | 00h | Size:       | 8 bits |
| Lockable:       | No  | Power Well: | Core   |

| Bit | Description                         |
|-----|-------------------------------------|
| 7:0 | <b>Programming Interface Value.</b> |

**13.1.7 SCC—Sub Class Code Register (Modem—D31:F6)**

|                 |     |             |        |
|-----------------|-----|-------------|--------|
| Address Offset: | 0Ah | Attribute:  | RO     |
| Default Value:  | 03h | Size:       | 8 bits |
| Lockable:       | No  | Power Well: | Core   |

| Bit | Description  |
|-----|--|
| 7:0 | <b>Sub Class Code Value.</b><br>03h = Generic Modem. |

**13.1.8 BCC—Base Class Code Register (Modem—D31:F6)**

|                 |     |             |        |
|-----------------|-----|-------------|--------|
| Address Offset: | 0Bh | Attribute:  | RO     |
| Default Value:  | 07h | Size:       | 8 bits |
| Lockable:       | No  | Power Well: | Core   |

| Bit | Description  |
|-----|--|
| 7:0 | <b>Base Class Code Value.</b><br>07h = Simple Communications Controller. |

**13.1.9 HEDTYP—Header Type Register (Modem—D31:F6)**

|                 |     |             |        |
|-----------------|-----|-------------|--------|
| Address Offset: | 0Eh | Attribute:  | RO     |
| Default Value:  | 00h | Size:       | 8 bits |
| Lockable:       | No  | Power Well: | Core   |

| Bit | Description                             |
|-----|---|
| 7:0 | <b>Header Value.</b> Always reads as 0. |



### 13.1.10 MMBAR—Modem Mixer Base Address Register (Modem—D31:F6)

|                 |           |            |         |
|-----------------|-----------|------------|---------|
| Address Offset: | 10h–13h   | Attribute: | R/W     |
| Default Value:  | 00000001h | Size:      | 32 bits |

The Native PCI Mode Modem uses PCI Base Address register #1 to request a contiguous block of I/O space that is to be used for the Modem Mixer software interface. The mixer requires 256 bytes of I/O space. All accesses to the mixer registers are forwarded over the AC-link to the codec where the registers reside.

In the case of the split codec implementation accesses to the different codecs are differentiated by the controller by using address offsets 00h–7Fh for the primary codec and address offsets 80h–FEh for the secondary codec.

| Bit   | Description   |
|-------|---|
| 31:16 | Hardwired to 0's  |
| 15:8  | <b>Base Address.</b> These bits are used in the I/O space decode of the Modem interface registers. The number of upper bits that a device actually implements depends on how much of the address space the device will respond to. For the AC '97 Modem, the upper 16 bits are hardwired to 0, while bits 15:8 are programmable. This configuration yields a maximum I/O block size of 256 bytes for this base address. |
| 7:1   | Reserved. Read as 0   |
| 0     | <b>Resource Type Indicator (RTE).</b> This bit is set to one, indicating a request for I/O space.   |

### 13.1.11 MBAR—Modem Base Address Register (Modem—D31:F6)

|                 |           |            |         |
|-----------------|-----------|------------|---------|
| Address Offset: | 14h–17h   | Attribute: | R/W     |
| Default Value:  | 00000001h | Size:      | 32 bits |

The Modem function uses PCI Base Address register #1 to request a contiguous block of I/O space that is to be used for the Modem software interface. The Modem Bus Mastering register space requires 128 bytes of I/O space. All Modem registers reside in the controller, therefore cycles are NOT forwarded over the AC-link to the codec.

| Bit   | Description   |
|-------|---|
| 31:16 | Hardwired to 0's  |
| 15:7  | <b>Base Address.</b> These bits are used in the I/O space decode of the Modem interface registers. The number of upper bits that a device actually implements depends on how much of the address space the device will respond to. For the AC '97 Modem, the upper 16 bits are hardwired to 0, while bits 15:7 are programmable. This configuration yields a maximum I/O block size of 128 bytes for this base address. |
| 6:1   | Reserved. Read as 0   |
| 0     | <b>Resource Type Indicator (RTE).</b> This bit is set to one, indicating a request for I/O space.   |

**13.1.12 SVID—Subsystem Vendor ID (Modem—D31:F6)**

|                 |         |             |            |
|-----------------|---------|-------------|------------|
| Address Offset: | 2Dh–2Ch | Attribute:  | Write-Once |
| Default Value:  | 0000h   | Size:       | 16 bits    |
| Lockable:       | No      | Power Well: | Core       |

This register should be implemented for any function that could be instantiated more than once in a given system, for example, a system with 2 audio subsystems, one down on the motherboard and the other plugged into a PCI expansion slot, should have the SVID register implemented. The SVID register, in combination with the Subsystem ID register, enable the operating environment to distinguish one audio subsystem from the other(s).

This register is implemented as write-once register. Once a value is written to it, the value can be read back. Any subsequent writes will have no effect.

| Bit  | Description                       |
|------|-----------------------------------|
| 15:0 | <b>Subsystem Vendor ID Value.</b> |

**13.1.13 SID—Subsystem ID (Modem—D31:F6)**

|                 |         |             |            |
|-----------------|---------|-------------|------------|
| Address Offset: | 2Fh–2Eh | Attribute:  | Write-Once |
| Default Value:  | 0000h   | Size:       | 16 bits    |
| Lockable:       | No      | Power Well: | Core       |

This register should be implemented for any function that could be instantiated more than once in a given system, for example, a system with 2 audio subsystems, one down on the motherboard and the other plugged into a PCI expansion slot. The SID register, in combination with the Subsystem Vendor ID register make it possible for the operating environment to distinguish between multiple audio subsystems in a single platform.

This register is implemented as write-once register. Once a value is written to it, the value can be read back. Any subsequent writes will have no effect.

| Bit  | Description                |
|------|----------------------------|
| 15:0 | <b>Subsystem ID Value.</b> |

**13.1.14 INTR\_LN—Interrupt Line Register (Modem—D31:F6)**

|                 |     |             |        |
|-----------------|-----|-------------|--------|
| Address Offset: | 3Ch | Attribute:  | R/W    |
| Default Value:  | 00h | Size:       | 8 bits |
| Lockable:       | No  | Power Well: | Core   |

This register indicates which PCI interrupt line is used for the AC'97 module interrupt.

| Bit | Description   |
|-----|---|
| 7:0 | <b>Interrupt Line:</b> This data is not used by the ICH. It is used to communicate to software the interrupt line that the interrupt pin is connected to. |



### 13.1.15 INT\_PIN—Interrupt Pin (Modem—D31:F6)

|                 |     |             |        |
|-----------------|-----|-------------|--------|
| Address Offset: | 3Dh | Attribute:  | RO     |
| Default Value:  | 02h | Size:       | 8 bits |
| Lockable:       | No  | Power Well: | Core   |

This register indicates which PCI interrupt pin is used for the AC '97 modem interrupt. The AC '97 interrupt is internally ORed to the interrupt controller with the PIRQB# signal.

| Bit | Description  |
|-----|--|
| 7:3 | Reserved.  |
| 2:0 | <b>AC '97 Interrupt Routing.</b> Hardwired to 010b to select PIRQB#. |

## 13.2 AC'97 Modem I/O Space (D31:F6)

In the case of the split codec implementation accesses to the modem mixer registers in different codecs are differentiated by the controller by using address offsets 00h–7Fh for the primary codec and address offsets 80h–FEh for the secondary codec. Table 13-2 shows the register addresses for the modem mixer registers.

**Table 13-2. ICH Modem Mixer Register Configuration**

| Register   |            | MMBAR Exposed Registers (D31:F6)  |
|------------|------------|-----------------------------------|
| Pri.       | Sec.       | Name                              |
| 00h:38h    | 80h:B8h    | Intel RESERVED                    |
| 3Ch        | BCh        | Extended Modem ID                 |
| 3Eh        | BEh        | Extended Modem Stat/Ctrl          |
| 40h        | C0h        | Line 1 DAC/ADC Rate               |
| <i>42h</i> | <i>C2h</i> | <i>Line 2 DAC/ADC Rate</i>        |
| <i>44h</i> | <i>C4h</i> | <i>Handset DAC/ADC Rate</i>       |
| 46h        | C6h        | Line 1 DAC/ADC Level Mute         |
| <i>48h</i> | <i>C8h</i> | <i>Line 2 DAC/ADC Level Mute</i>  |
| <i>4Ah</i> | <i>CAh</i> | <i>Handset DAC/ADC Level Mute</i> |
| 4Ch        | CCh        | GPIO Pin Config                   |
| 4Eh        | CEh        | GPIO Polarity/Type                |
| 50h        | D0h        | GPIO Pin Sticky                   |
| 52h        | D2h        | GPIO Pin Wake Up                  |
| 54h        | D4h        | GPIO Pin Status                   |
| 56h        | D6h        | Misc. Modem AFE Stat/Ctrl         |
| <b>58h</b> | <b>D8h</b> | <b>Vendor Reserved</b>            |
| <b>7Ah</b> | <b>FAh</b> | <b>Vendor Reserved</b>            |
| <b>7Ch</b> | <b>FCh</b> | <b>Vendor ID1</b>                 |
| <b>7Eh</b> | <b>FEh</b> | <b>Vendor ID2</b>                 |

**NOTE:**

1. Registers in bold are multiplexed between audio and modem functions
2. Registers in italics are for functions not supported by the ICH
3. Software should not try to access reserved registers
4. The ICH supports a modem codec as either primary or secondary, but does not support two modem codecs.

The Global Control (GLOB\_CNT) and Global Status (GLOB\_STA) registers are aliased to the same global registers in the audio and modem I/O space. Therefore a read/write to these registers in either audio or modem I/O space affects the same physical register.

These registers exist in I/O space and reside in the AC '97 controller. The two channels, Modem in and Modem out, each have their own set of Bus Mastering registers. The following register descriptions apply to both channels. The naming prefix convention used is as follows:

MI = Modem in channel

MO = Modem out channel

**Table 13-3. Modem Registers**

| Offset | Mnemonic | Name   | Default   | Access |
|--------|----------|--|-----------|--------|
| 00h    | MI_BDBAR | Modem In Buffer Descriptor List Base Address Register  | 00000000h | R/W    |
| 04h    | MI_CIV   | Modem In Current Index Value Register                  | 00h       | R      |
| 05h    | MI_LVI   | Modem In Last Valid Index Register                     | 00h       | R/W    |
| 06h    | MI_SR    | Modem In Status Register                               | 0001h     | R/W    |
| 08h    | MI_PICB  | Modem In Position In Current Buffer Register           | 00h       | R      |
| 0Ah    | MI_PIV   | Modem In Prefetch Index Value Register                 | 00h       | RO     |
| 0Bh    | MI_CR    | Modem In Control Register                              | 00h       | R/W    |
| 10h    | MO_BDBAR | Modem Out Buffer Descriptor List Base Address Register | 00000000h | R/W    |
| 14h    | MO_CIV   | Modem Out Current Index Value Register                 | 00h       | RO     |
| 15h    | MO_LVI   | Modem Out Last Valid Register                          | 00h       | R/W    |
| 16h    | MO_SR    | Modem Out Status Register                              | 0001h     | R/W    |
| 18h    | MI_PICB  | Modem In Position In Current Buffer Register           | 00h       | RO     |
| 1Ah    | MO_PIV   | Modem Out Prefetched Index Register                    | 00h       | RO     |
| 1Bh    | MO_CR    | Modem Out Control Register                             | 00h       | R/W    |
| 3Ch    | GLOB_CNT | Global Control   | 00000000h | R/W    |
| 40h    | GLOB_STA | Global Status  | 00000000h | RO     |
| 44h    | ACC_SEMA | Codec Write Semaphore Register                         | 00h       | R/W    |

**NOTE:**

1. MI = Modem in channel; MO = Modem out channel

**13.2.1 x\_BDBAR—Buffer Descriptor List Base Address Register**

I/O Address: MBAR + 00h (MIBDBAR), Attribute: R/W  
 MBAR + 10h (MOBDBAR)  
 Default Value: 00000000h Size: 32bits  
 Lockable: No Power Well: Core

| Bit  | Description  |
|------|--|
| 31:3 | <b>Buffer Descriptor List Base address[31:3]</b> . These bits represent address bits 31:3. The entries should be aligned on 8 byte boundaries. |
| 2:0  | Hardwired to 0.  |

**13.2.2 x\_CIV—Current Index Value Register**

I/O Address: MBAR + 04h (MICIV), Attribute: RO  
 MBAR + 14h (MOCIV),  
 Default Value: 00h Size: 8bits  
 Lockable: No Power Well: Core

| Bit | Description   |
|-----|---|
| 7:5 | Hardwired to 0.   |
| 4:0 | <b>Current Index Value [4:0]</b> These bits represent which buffer descriptor within the list of 16 descriptors is being processed currently. As each descriptor is processed, this value is incremented. |

**13.2.3 x\_LVI—Last Valid Index Register**

I/O Address: MBAR + 05h (MILVI), Attribute: R/W  
 MBAR + 15h (MOLVI)  
 Default Value: 00h Power Well: Core

| Bit | Description   |
|-----|---|
| 7:5 | Hardwired to 0  |
| 4:0 | <b>Last Valid Index [4:0]</b> These bits indicate the last valid descriptor in the list. This value is updated by the software as it prepares new buffers and adds to the list. |

### 13.2.4 x\_SR—Status Register

|                |   |             |         |
|----------------|---|-------------|---------|
| I/O Address:   | MBAR + 06h (MISR),<br>MBAR + 16h (MOSR) | Attribute:  | R/W     |
| Default Value: | 0001h                                   | Size:       | 16 bits |
| Lockable:      | No                                      | Power Well: | Core    |

| Bit  | Description  |
|------|--|
| 15:5 | Reserved.  |
| 4    | <p><b>FIFO error (FIFOE).</b> The ICH will set the FIFOE bit if the under-run or over-run occurs when there are more valid buffers to process.</p> <p>1 = FIFO error occurs.<br/>0 = Cleared by writing a "1" to this bit position.</p> <p><b>NOTES:</b></p> <ol style="list-style-type: none"> <li><b>Modem in:</b> FIFO error indicates a FIFO overrun. The FIFO pointers don't increment, the incoming data is not written into the FIFO, thus is lost.</li> <li><b>Modem out:</b> FIFO error indicates a FIFO under-run. The sample transmitted in this case should be the last valid sample.</li> </ol>   |
| 3    | <p><b>Buffer Completion Interrupt Status (BCIS)</b></p> <p>1 = Set by the hardware after the last sample of a buffer has been processed, AND if the Interrupt on Completion (IOC) bit is set in the command byte of the buffer descriptor. Remains active until software clears bit.</p> <p>0 = Cleared by writing a "1" to this bit position.</p>   |
| 2    | <p><b>Last Valid Buffer Completion Interrupt (LVBCI)</b></p> <p>1 = Set by hardware when last valid buffer has been processed. It remains active until cleared by software. This bit indicates the occurrence of the event signified by the last valid buffer being processed. Thus, this is an event status bit that can be cleared by software once this event has been recognized. This event will cause an interrupt if the enable bit in the Control Register is set. The interrupt is cleared when the software clears this bit.</p> <p>In the case of transmits (PCM out, Modem out) this bit is set, after the last valid buffer has been fetched (not after transmitting it) While in the case of Receives, this bit is set after the data for the last buffer has been written to memory.</p> <p>0 = cleared by writing a "1" to this bit position</p> |
| 1    | <p><b>Current Equals Last Valid (CELV)</b></p> <p>1 = Current Index is equal to the value in the Last Valid Index Register, AND the buffer pointed to by the CIV has been processed (i.e., after the last valid buffer has been processed). This bit is very similar to bit 2, except, this bit reflects the state rather than the event. This bit reflects the state of the controller, and remains set until the controller exits this state.</p> <p>0 = Hardware clears when controller exists state (i.e., until a new value is written to the LVI register).</p>  |
| 0    | <p><b>DMA Controller Halted (DCH)</b></p> <p>1 = DMA controller is halted. This could happen because of the Start/Stop bit being cleared, or it could happen once the controller has processed the last valid buffer (in which case it will set bit 1 and halt).</p>   |

### 13.2.5 x\_PICB—Position In Current Buffer Register

I/O Address: MBAR + 08h (MIPICB), Attribute: RO  
 MBAR + 18h (MOPICB),  
 Default Value: 0000h Size: 16 bits  
 Lockable: No Power Well: Core

| Bit  | Description   |
|------|---|
| 15:0 | <b>Position In Current Buffer[15:0]</b> These bits represent the number of dwords left to be processed in the current buffer. |

### 13.2.6 x\_PIV—Prefetch Index Value Register

I/O Address: MBAR + 0Ah (MIPIV), Attribute: RO  
 MBAR + 1Ah (MOPIV)  
 Default Value: 00h Size: 8 bits  
 Lockable: No Power Well: Core

| Bit | Description   |
|-----|---|
| 7:5 | Hardwired to 0  |
| 4:0 | <b>Prefetched Index value [4:0]</b> These bits represent which buffer descriptor in the list has been prefetched. |



### 13.2.7 x\_CR—Control Register

|                |   |             |        |
|----------------|---|-------------|--------|
| I/O Address:   | MBAR + 0Bh (MICR),<br>MBAR + 1Bh (MOCR) | Attribute:  | R/W    |
| Default Value: | 00h                                     | Size:       | 8 bits |
| Lockable:      | No                                      | Power Well: | Core   |

| Bit | Description  |
|-----|--|
| 7:5 | Reserved.  |
| 4   | <b>Interrupt On Completion Enable (IOCE).</b> This bit controls whether or not an interrupt occurs when a buffer completes with the IOC bit set in its descriptor.   |
| 3   | <b>FIFO Error Interrupt Enable (FEIE).</b> This bit controls whether the occurrence of a FIFO error will cause an interrupt or not.<br>1 = Enable. Interrupt will occur<br>0 = Disable. Bit 4 in the Status Register will be set, but the interrupt will not occur.  |
| 2   | <b>Last Valid Buffer Interrupt Enable (LVBIE).</b> This bit controls whether the completion of the last valid buffer will cause an interrupt or not.<br>1 = Enable.<br>0 = Disable. Bit 2 in the Status register will still be set, but the interrupt will not occur.  |
| 1   | <b>Reset Registers(RR).</b><br>1 = Contents of all registers to be reset, except the interrupt enable bits (bit 4,3,2 of this register). Software needs to set this bit. It must be set only when the Run/Pause bit is cleared. Setting it when the Run bit is set will cause undefined consequences. This bit is self-clearing (software needs not clear it).<br>0 = Removes reset condition. |
| 0   | <b>Run/Pause Bus master (RPBM).</b><br>1=Run. Bus master operation starts.<br>0 = Pause bus master operation. This results in all state information being retained (i.e., master mode operation can be stopped and then resumed).  |

## 13.2.8 GLOB\_CNT—Global Control Register

|                |            |             |         |
|----------------|------------|-------------|---------|
| I/O Address:   | MBAR + 3Ch | Attribute:  | R/W     |
| Default Value: | 00000000h  | Size:       | 32 bits |
| Lockable:      | No         | Power Well: | Core    |

| Bit  | Description  |
|------|--|
| 31:6 | Reserved.  |
| 5    | <b>Secondary Resume Interrupt Enable</b><br>1 = Enable an interrupt to occur when the secondary codec causes a resume event on the AC-link.<br>0 = Disable   |
| 4    | <b>Primary Resume Interrupt Enable</b><br>1 = Enable an interrupt to occur when the primary codec causes a resume event on the AC-link.<br>0 = Disable   |
| 3    | <b>ACLINK Shut Off</b><br>1 = Disable the AC-link signals (drive all AC'97 outputs low and turn off all AC'97 input buffer enables)  |
| 2    | <b>AC'97 Warm Reset.</b><br>1 = Writing a "1" to this bit causes a warm reset to occur on the AC-link. The warm reset will awaken a suspended codec without clearing its internal registers. If software attempts to perform a warm reset while BIT_CLK is running, the write will be ignored and the bit will not be changed. A warm reset can only occur in the absence of BIT_CLK. This bit is self-clearing (it clears itself after the reset has occurred and BIT_CLK has started). |
| 1    | <b>AC'97 Cold Reset#.</b><br>0 = Writing a "0" to this bit causes a cold reset to occur throughout the AC'97 circuitry. All data in the codec will be lost. Software needs to clear this bit no sooner than after 1usec has elapsed. This bit reflects the state of the AC_RST# pin. The ICH clears this bit to "0" upon entering S3/S4/S5 sleep states and PCIRST#.   |
| 0    | <b>GPI Interrupt Enable (GIE).</b> This bit controls whether the change in status of any GPI causes an interrupt.<br>1 = The change on value of a GPI causes an interrupt and sets bit 0 of the Global Status Register.<br>0 = Bit 0 of the Global Status Register is set, but an interrupt is not generated.  |

## 13.2.9 GLOB\_STA—Global Status Register

|                |            |             |         |
|----------------|------------|-------------|---------|
| I/O Address:   | MBAR + 40h | Attribute:  | R/W     |
| Default Value: | 00000000h  | Size:       | 32 bits |
| Lockable:      | No         | Power Well: | Core    |

| Bit   | Description  |
|-------|--|
| 31:18 | Reserved.  |
| 17    | <b>MD3:</b> Power down semaphore for Modem. This bit is used by software in conjunction with the AD3 bit to coordinate the entry of the two codecs into D3 state. This bit resides in the resume well and maintains context across power states. |
| 16    | <b>AD3:</b> Power down semaphore for Audio. This bit is used by software in conjunction with the MD3 bit to coordinate the entry of the two codecs into D3 state. This bit resides in the resume well and maintains context across power states. |
| 15    | <b>Read Completion Status:</b> This bit indicates the status of Codec read completions.<br>1 = A Codec read results in a time-out. This bit remains set until being cleared by software.<br>0 = A Codec read completes normally.                 |
| 14    | <b>Bit 3 of slot 12:</b> Display bit 3 of the most recent slot 12  |

| Bit | Description  |
|-----|--|
| 13  | <b>Bit 2 of slot 12:</b> Display bit 2 of the most recent slot 12  |
| 12  | <b>Bit 1 of slot 12:</b> Display bit 1 of the most recent slot 12  |
| 11  | <b>Secondary Resume Interrupt.</b> This bit indicates that a resume event occurred on SIDN[1].<br>1 = Resume event occurred<br>0 = Cleared by writing a 1 to this bit position.  |
| 10  | <b>Primary Resume Interrupt.</b> This bit indicates that a resume event occurred on SDIN[0].<br>1 = Resume event occurred<br>0 = Cleared by writing a 1 to this bit position.  |
| 9   | <b>Secondary Codec Ready (SCR).</b> Reflects the state of the codec ready bit in SDIN[1]. Bus masters ignore the condition of the codec ready bits. Software must check this bit before starting the bus masters. Once the codec is "ready", it must never go "not ready" spontaneously. |
| 8   | <b>Primary Codec Ready (PCR).</b> Reflects the state of the codec ready bit in SDIN[0]. Bus masters ignore the condition of the codec ready bits. Software must check this bit before starting the bus masters. Once the codec is "ready", it must never go "not ready" spontaneously.   |
| 7   | <b>Mic In Interrupt (MINT).</b> This bit indicates that one of the Mic in channel interrupts occurred.<br>1 = Interrupt occurred.<br>0 = When the specific interrupt is cleared, this bit will be cleared.   |
| 6   | <b>PCM Out Interrupt (POINT).</b> This bit indicates that one of the PCM out channel interrupts occurred.<br>1 = Interrupt occurred.<br>0 = When the specific interrupt is cleared, this bit will be cleared.  |
| 5   | <b>PCM In Interrupt (PIINT).</b> This bit indicates that one of the PCM in channel interrupts occurred.<br>1 = Interrupt occurred.<br>0 = When the specific interrupt is cleared, this bit will be cleared.  |
| 4:3 | Reserved   |
| 2   | <b>Modem Out Interrupt (MOINT).</b> This bit indicates that one of the modem out channel interrupts occurred.<br>1 = Interrupt occurred.<br>0 = When the specific interrupt is cleared, this bit will be cleared.  |
| 1   | <b>Modem In Interrupt (MIINT).</b> This bit indicates that one of the modem in channel interrupts occurred.<br>1 = Interrupt occurred.<br>0 = When the specific interrupt is cleared, this bit will be cleared.  |
| 0   | <b>GPI Status Change Interrupt (GSCI).</b> This bit is set whenever bit 0 of slot 12 is set. This happens when the value of any of the GPIOs currently defined as inputs changes.<br>1 = Input changed.<br>0 = Cleared by writing a 1 to this bit position.                              |

**Note:** On reads from a codec, the controller will give the codec a maximum of 4 frames to respond, after which if no response is received, it will return a dummy read completion to the processor (with all F's on the data) and also set the Read Completion Status bit in the Global Status Register.

### 13.2.10 CAS—Codec Access Semaphore Register

|                |               |             |        |
|----------------|---------------|-------------|--------|
| I/O Address:   | NABMBAR + 44h | Attribute:  | R/W    |
| Default Value: | 00h           | Size:       | 8 bits |
| Lockable:      | No            | Power Well: | Core   |

| Bit | Description  |
|-----|--|
| 7:1 | Reserved.  |
| 0   | <p><b>Codec Access Semaphore (CAS):</b> This bit is read by software to check whether a codec access is currently in progress.</p> <p>1 = The act of reading this register sets this bit to 1. The driver that read this bit can then perform an I/O access. Once the access is completed H/W automatically clears this bit.</p> <p>0 = No access in progress.</p> |

# Electrical Characteristics

## 14.1 Absolute Maximum Ratings

|  |                                  |
|--|----------------------------------|
| Case Temperature under Bias . . . . .  | 0°C to +100°C                    |
| Storage Temperature . . . . .  | -55°C to +150°C                  |
| Voltage on Any 3.3V Pin with Respect to Ground . . . . .                               | -0.5V to V <sub>CC</sub> + 0.3V  |
| Voltage on Any 5V Tolerant Pin with Respect to Ground (V <sub>REF</sub> =5V) . . . . . | -0.5V to V <sub>REF</sub> + 0.3V |
| 1.8V Supply Voltage with Respect to V <sub>SS</sub> . . . . .                          | -0.3V to V <sub>REF</sub> + 2.7V |
| 3.3V Supply Voltage with Respect to V <sub>SS</sub> . . . . .                          | -0.5V to +4.6V                   |
| 5.0V Supply Voltage (V <sub>ref</sub> ) with Respect to V <sub>SS</sub> . . . . .      | -0.5V to +5.5V                   |
| Maximum Power Dissipation . . . . .  | 1.4 W                            |

**Warning:** Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operating beyond the "Operating Conditions" is not recommended and extended exposure beyond "Operating Conditions" may affect reliability.

## 14.2 Thermal Characteristics

The ICH is designed for operation at case temperatures between 0°C and 100°C.

## 14.3 D.C. Characteristics

Table 14-1. D.C. Current Characteristics

| Functional Operating Range (V <sub>REF</sub> = 5V ±5%, V <sub>CC</sub> = 3.3v ±0.3V, T <sub>CASE</sub> = 0°C to +100°C) |   |     |     |      |       |
|---|---|-----|-----|------|-------|
| Symbol  | Parameter   | Typ | Max | Unit | Notes |
| I <sub>CC</sub> (1.8V)  | Hub Interface Supply Current                              |     | 55  | mA   |       |
| I <sub>CC</sub> (3V)  | V <sub>CC</sub> Supply Current                            |     | 300 | mA   |       |
| I <sub>CC</sub> (SUS)-ON  | Suspend Well Supply Current - Full On                     | 30  | 100 | mA   |       |
| I <sub>CC</sub> (SUS)-STR   | Suspend Well Supply Current - Suspend to RAM              | 0.8 | 2   | mA   |       |
| I <sub>CC</sub> (SUS)-STD/Soff  | Suspend Well Supply Current - Suspend to Disk or Soft Off | 0.8 | 2   | mA   |       |
| I <sub>CC</sub> (RTC)   | Battery Standby Current                                   | 1.7 | 3.5 | uA   | 1     |

**NOTES:**  
 1. I<sub>CC</sub>(RTC) data is taken with V<sub>CC</sub>(RTC) at 3.0V while the system is in a mechanical off (G3) state at room temperature. As V<sub>CC</sub>(RTC) drops the current will also drop. At 2.5V the typical I<sub>CC</sub>(RTC) is 1.4uA.

Table 14-2. DC Characteristic Input Signal Association

| Symbol                                      | Associated Signals  |
|---|---|
| VIH1/VIL1<br>(5V Tolerant)                  | <b>ICH (82801AA) and ICH0 (82801AB):</b>  |
|   | <b>IDE Signals:</b> PDD[15:0], SDD[15:0], PDDREQ, PIORDY, SDDREQ, SIORDY  |
|   | <b>PCI Signals:</b> AD[31:0], C/BE[3:0]#, GPIO1/REQB#, GPIO0/REQA#, REQ[3:0]#, PAR, DEVSEL#, STOP#, TRDY#, SERR#, IRDY#, FRAME# |
|   | <b>Interrupt Signals:</b> IRQ[15:14], SERIRQ, PIRQ[A:D]#<br><b>Legacy Signals:</b> RCIN#, A20GATE                               |
|   | <b>ICH (82801AA) Only:</b>  |
|   | <b>PCI Signals:</b> GPIO1/REQB#/REQ5#, REQ[4]#, GPIO7/PERR#   |
|   | <b>ICH0 (82801AB) Only:</b>   |
|   | <b>General Purpose Signals:</b> GPIO[7], GPIO1/REQB#  |
| VIH2/VIL2                                   | <b>Processor Signals:</b> FERR#, APICD[1:0]   |
| VIH3/VIL3                                   | <b>PCI Signals:</b> PME#  |
|   | <b>Power Signals:</b> PWRBTN#   |
|   | <b>LPC Signals:</b> GPIO8/LDRQ1#, LAD[3:0], LDRQ0#  |
|   | <b>Power Management Signals:</b> RSMRST#, RTCRST#, RI#, PWROK, THRM#  |
|   | <b>SMBus Signals:</b> GPIO11/SMBALERT#  |
|   | <b>System Management Signals:</b> GPIO10/INTRUDER#  |
|   | <b>USB Signals:</b> OC[1:0]#  |
| <b>General Purpose Signals:</b> GPIO[13:12] |   |
| VIH4/VIL4                                   | <b>AC'97 Signals:</b> GPIO9/AC_SDIN1, AC_SDIN0, AC_BITCLK   |
| VIL5/VIH5                                   | <b>ICH (82801AA) and ICH0 (82801AB):</b>  |
|   | <b>SMBus Signals:</b> SMBCLK, SMBDATA   |
|   | <b>ICH (82801AA) Only:</b>  |
|   | <b>SMBus Signals:</b> GPIO27/ALERTCLK,GPIO28/ALERTDATA  |
|   | <b>ICH0 (82801AB) Only:</b>   |
|   | <b>General Purpose Signals:</b> GPIO[28:27]   |
| VIL7/VIH7                                   | <b>General Purpose Signals:</b> GPIO[6:5]   |

Table 14-3. DC Input Characteristics

| Symbol           | Parameter          | Min                 | Max                   | Unit | Notes |
|------------------|--------------------|---------------------|-----------------------|------|-------|
| V <sub>IL1</sub> | Input Low Voltage  | -0.5                | 0.8                   | V    |       |
| V <sub>IH1</sub> | Input High Voltage | 2                   | V <sub>cc5</sub> +0.5 | V    |       |
| V <sub>IL2</sub> | Input Low Voltage  | -0.5                | 0.6                   | V    |       |
| V <sub>IH2</sub> | Input High Voltage | 1.2                 | V <sub>cc</sub> +0.5  | V    |       |
| V <sub>IL3</sub> | Input Low Voltage  | -0.5                | 0.3V <sub>cc</sub>    | V    |       |
| V <sub>IH3</sub> | Input High Voltage | 0.5V <sub>cc</sub>  | V <sub>cc</sub> +0.5  | V    |       |
| V <sub>IL4</sub> | Input Low Voltage  | -0.5                | 0.35V <sub>cc</sub>   | V    |       |
| V <sub>IH4</sub> | Input High Voltage | 0.65V <sub>cc</sub> | V <sub>cc</sub> +0.5  | V    |       |
| V <sub>IL5</sub> | Input Low Voltage  | -0.5                | 0.6                   | V    |       |
| V <sub>IH5</sub> | Input High Voltage | 1.4                 | V <sub>cc</sub> +0.3  | V    |       |
| V <sub>IL7</sub> | Input Low Voltage  | -0.5                | 0.8                   | V    |       |
| V <sub>IH7</sub> | Input High Voltage | 2                   | V <sub>cc</sub> +0.5  | V    |       |

Table 14-4. DC Characteristic Output Signal Association (Sheet 1 of 2)

| Symbol    | Associated Signals  |
|-----------|---|
| VOH1/VOL1 | <b>IDE Signals:</b> PDD[15:0], PDIOW#, PDIOR#, PDDACK#, PDA[2:0], PDCS1#, PDCS3#, SDD[15:0], SDDACK#, SDIOW#, SDIOR#, SDA[2:0], SDCS1#, SDCS3#  |
| VOH2/VOL2 | <b>Processor Signals:</b> SMI#, IGNNE#, INIT#, STPCLK#, INTR, A20M#, NMI, CPUSLP#<br><b>General Purpose Signals:</b> GPIO23   |
| VOH3/VOL3 | <b>ICH (82801AA) and ICH0 (82801AB):</b><br><b>PCI Signals:</b> GPIO17/GNTB#, GPIO16/GNTA#<br><b>LPC Signals:</b> LAD[3:0], LFRAME#<br><b>AC'97 Signals:</b> AC_RST#, AC_SDOUT, AC_SYNC                     |
|           | <b>ICH (82801AA) Only:</b><br><b>PCI Signals:</b> GPIO17/GNTB#/GNT5#  |
| VOH5/VOL5 | <b>ICH (82801AA) and ICH0 (82801AB):</b><br><b>PCI Signals:</b> AD[31:0], C/BE[3:0]#, PCIRST#, GNT[4:0]#, PAR, DEVSEL#, PLOCK#, STOP#, TRDY#, IRDY#, FRAME#<br><b>Interrupt Signals:</b> SERIRQ, PIRQ[A:D]# |
|           | <b>ICH (82801AA) Only:</b><br><b>PCI Signals:</b> GPIO7/PERR#   |
|           | <b>ICH0 (82801AB) Only:</b><br><b>General Purpose Signals:</b> GPIO[7]  |

Table 14-4. DC Characteristic Output Signal Association (Sheet 2 of 2)

| Symbol    | Associated Signals  |
|-----------|---|
| VOL6/VOH6 | <b>ICH (82801AA) and ICH0 (82801AB):</b><br><b>SMBus Signals:</b> SMBCLK, SMBDATA<br><b>Power Management Signals:</b> GPIO24/SLP_S3#, GPIO25/SUSSTAT#, SLP_S5#<br><b>General Purpose Signals:</b> GPIO[22:21]<br><b>Other Signals:</b> SPKR |
|           | <b>ICH (82801AA) Only:</b><br><b>SMBus Signals:</b> GPIO27/ALERTCLK, GPIO28/ALERTDATA<br><b>Power Management Signals:</b> GPIO26/SUSCLK   |
|           | <b>ICH0 (82801AB) Only:</b><br><b>General Purpose Signals:</b> GPIO[28:26]  |
| VOL7/VOH7 | <b>USB Signals:</b> USBPO[P:N], USBP1[P:N]  |
| VOL9/VOH9 | <b>Interrupt Signals:</b> APICD[1:0]  |

Table 14-5. DC Output Characteristics

| Symbol | Parameter           | Min                   | Max                | Unit | Notes                          |
|--------|---------------------|-----------------------|--------------------|------|--------------------------------|
| VOL1   | Output Low Voltage  |                       | 0.5                | V    | IOL1=4mA                       |
| VOH1   | Output High Voltage | 2.4                   |                    | V    | I <sub>OH1</sub> =-1mA         |
| VOL2   | Output Low Voltage  |                       | 0.4                | V    | IOL2=10mA                      |
| VOH2   | Output High Voltage |                       | 2.625              | V    | Note 1                         |
| VOL3   | Output Low Voltage  |                       | 0.1V <sub>cc</sub> | V    | IOL3=1.5mA                     |
| VOH3   | Output High Voltage | 0.9V <sub>cc</sub>    |                    | V    | I <sub>OH3</sub> =-0.5mA       |
| VOL5   | Output Low Voltage  |                       | 0.55               | V    | IOL5=6mA                       |
| VOH5   | Output High Voltage | 2.4                   |                    | V    | I <sub>OH5</sub> =-2mA, Note 2 |
| VOL6   | Output Low Voltage  |                       | 0.4                | V    | IOL6=3mA                       |
| VOH6   | Output High Voltage | V <sub>cc</sub> - 0.5 |                    | V    | I <sub>OH6</sub> =-2mA, Note 3 |
| VOL7   | Output Low Voltage  |                       | 0.4                | V    | IOL7=5mA                       |
| VOH7   | Output High Voltage | V <sub>cc</sub> - 0.5 |                    | V    | I <sub>OH7</sub> =-2mA         |
| VOL9   | Output Low Voltage  |                       | .45                | V    | IOL9=14mA                      |
| VOH9   | Output High Voltage |                       | 2.625              | V    | Note 1                         |

**NOTES:**

- All output signals associated with VOH2 are open drain drivers, which is why they have no VOH min spec. These signals must have external pull up resistors to a voltage no greater than VOH2 max.
- Interrupt Signals in VOH5 are open drain drivers, and the VOH min spec does not apply. These signals must have external pull up resistors.
- SMBUS Signals in VOH6 are open drain drivers, and the VOH min spec does not apply. These signals must have external pull up resistors.



Table 14-6. Other DC Characteristics

| Symbol           | Parameter  | Min          | Max          | Unit | Notes                                  |
|------------------|--|--------------|--------------|------|--|
| VREF             | ICH reference Voltage  | 4.75         | 5.25         | V    |  |
| Vcc              | Core Well Voltage  | 3.0          | 3.6          | V    |  |
| Vcc1.8           | Hub Link Voltage   | 1.7          | 1.9          | V    |  |
| HUBREF           | Hub Link Reference Voltage                                   | 0.48(Vcc1.8) | 0.52(Vcc1.8) | V    |  |
| Vcc(RTC)         | Battery Voltage  | 2.0          | 3.6          | V    |  |
| Vcc(SUS)         | Suspend Well Voltage   | 3.0          | 3.6          | V    |  |
| V <sub>IT+</sub> | Hysteresis Input Rising Threshold                            | 1.9          |              | V    | Applied to<br>USBP0[P:N]<br>USBP1[P:N] |
| V <sub>IT-</sub> | Hysteresis Input Falling Threshold                           |              | 1.3          | V    | Applied to<br>USBP0[P:N]<br>USBP1[P:N] |
| V <sub>DI</sub>  | Differential Input Sensitivity                               | 0.2          |              | V    | {(USBPx+,USBPx-)}                      |
| V <sub>CM</sub>  | Differential Common Mode Range                               | 0.8          | 2.5          | V    | Includes V <sub>DI</sub>               |
| V <sub>CRS</sub> | Output Signal Crossover Voltage                              | 1.3          | 2.0          | V    |  |
| V <sub>SE</sub>  | Single Ended Rcvr Threshold                                  | 0.8          | 2.0          | V    |  |
| I <sub>LI1</sub> | Input Leakage Current  |              | ±1           | uA   |  |
| I <sub>LI2</sub> | Hi-Z State Data Line Leakage                                 | -10          | +10          | uA   | (0 V < V <sub>IN</sub> < 3.3V)         |
| C <sub>IN</sub>  | Input Capacitance - Hublink<br>Input Capacitance - All Other |              | 8<br>12      | pF   | F <sub>C</sub> = 1 MHz                 |
| C <sub>OUT</sub> | Output Capacitance   |              | 12           | pF   | F <sub>C</sub> = 1 MHz                 |
| C <sub>I/O</sub> | I/O Capacitance  |              | 12           | pF   | F <sub>C</sub> = 1 MHz                 |
| C <sub>L</sub>   | Crystal Load Capacitance                                     | 7.5          | 15           | pF   |  |

## 14.4 A.C. Characteristics

Table 14-7. Clock Timings

| Functional Operating Range (V <sub>REF</sub> = 5V ±5%, V <sub>CC</sub> = 3.3V ±0.3V, T <sub>CASE</sub> = 0°C to +100°C) |                     |       |       |      |       |        |
|---|---------------------|-------|-------|------|-------|--------|
| Sym   | Parameter           | Min   | Max   | Unit | Notes | Figure |
| <b>PCI Clock (PCICLK)</b>   |                     |       |       |      |       |        |
| t1  | Period              | 30    | 33.3  | ns   |       | 15-2   |
| t2  | High Time           | 12    |       | ns   |       | 15-2   |
| t3  | Low Time            | 12    |       | ns   |       | 15-2   |
| t4  | Rise Time           |       | 3     | ns   |       | 15-2   |
| t5  | Fall Time           |       | 3     | ns   |       | 15-2   |
| <b>Oscillator Clock (OSC)</b>   |                     |       |       |      |       |        |
| t6  | Period              | 67    | 70    | ns   |       | 15-2   |
| t7  | High Time           | 20    |       |      |       | 15-2   |
| t8  | Low time            | 20    |       | ns   |       | 15-2   |
| <b>USB Clock (USBCLK)</b>   |                     |       |       |      |       |        |
| f <sub>clk48</sub>  | Operating Frequency | 48    |       | MHz  |       |        |
| t9  | Frequency Tolerance |       | 2500  | ppm  | 1     |        |
| t10   | High Time           | 7     |       | ns   |       | 15-2   |
| t11   | Low time            | 7     |       | ns   |       | 15-2   |
| t12   | Rise Time           |       | 1.2   | ns   |       | 15-2   |
| t13   | Fall Time           |       | 1.2   | ns   |       | 15-2   |
| <b>Suspend Clock (SUSCLK)<br/>(ICH 82801AA Only)</b>  |                     |       |       |      |       |        |
| f <sub>susclk</sub>   | Operating Frequency | 32    |       | KHz  | 5     |        |
| t14   | High time           | 10    |       | us   | 5     | 15-2   |
| t15   | Low Time            | 10    |       | us   | 5     | 15-2   |
| <b>SMBus Clock (SMBCLK)</b>   |                     |       |       |      |       |        |
| f <sub>smb</sub>  | Operating Frequency | 10    | 16    | KHz  |       |        |
| t18   | High time           | 4.0   | 50    | us   | 2     | 15-17  |
| t19   | Low time            | 4.7   |       | us   |       | 15-17  |
| t20   | Rise time           |       | 1000  | ns   |       | 15-17  |
| t21   | Fall time           |       | 300   | ns   |       | 15-17  |
| <b>I/O APIC Clock (APICCLK)</b>   |                     |       |       |      |       |        |
| f <sub>ioap</sub>   | Operating Frequency | 14.32 | 16.67 | MHz  |       |        |
| t22   | High time           | 12    | 36    | ns   |       | 15-2   |
| t23   | Low time            | 12    | 36    | ns   |       | 15-2   |
| t24   | Rise time           | 1.0   | 5.0   | ns   |       | 15-2   |

Table 14-7. Clock Timings

| Functional Operating Range (V <sub>REF</sub> = 5V ±5%, V <sub>CC</sub> = 3.3V ±0.3V, T <sub>CASE</sub> = 0°C to +100°C) |                     |        |       |      |       |        |
|---|---------------------|--------|-------|------|-------|--------|
| Sym   | Parameter           | Min    | Max   | Unit | Notes | Figure |
| t25   | Fall time           | 1.0    | 5.0   | ns   |       | 15-2   |
| <b>AC'97 Clock (BITCLK)</b>   |                     |        |       |      |       |        |
| fac97   | Operating Frequency | 12.288 |       | MHz  |       |        |
| t26   | Output Jitter       |        | 750   | ps   |       |        |
| t27   | High time           | 32.56  | 48.84 | ns   |       | 15-2   |
| t28   | Low time            | 32.56  | 48.84 | ns   |       | 15-2   |
| t29   | Rise time           | 2.0    | 6.0   | ns   | 4     | 15-2   |
| t30   | Fall time           | 2.0    | 6.0   | ns   | 4     | 15-2   |
| <b>Hub Interface Clock</b>  |                     |        |       |      |       |        |
| fhl   | Operating Frequency | 66     |       | MHz  |       |        |
| t31   | High time           | TBD    | TBD   | ns   |       | 15-2   |
| t32   | Low time            | TBD    | TBD   | ns   |       | 15-2   |
| t33   | Rise time           | TBD    | TBD   | ns   |       | 15-2   |
| t34   | Fall time           | TBD    | TBD   | ns   |       | 15-2   |
| t35   | CLK66 leads PCICLK  | 1.0    | 4.5   | ns   | 3     |        |

**NOTES:**

1. The USBCLK is a 48 MHz that expects a 40/60% duty cycle.
2. The maximum high time (t18 Max) provide a simple guaranteed method for devices to detect bus idle conditions.
3. This specification includes pin-to-pin skew from the clock generator as well as board skew.
4. BITCLK Rise and Fall times are measured from 10%VDD and 90%VDD.
5. For the ICH (82801AA) SUSCLK duty cycle can range from 30% minimum to 70% maximum.

Table 14-8. PCI Interface Timing

| Functional Operating Range ( $V_{REF} = 5V \pm 5\%$ , $V_{CC} = 3.3V \pm 0.3V$ , $T_{CASE} = 0^{\circ}C$ to $+100^{\circ}C$ ) |  |     |     |       |                       |        |
|---|--|-----|-----|-------|-----------------------|--------|
| Sym   | Parameter  | Min | Max | Units | Notes                 | Figure |
| t40   | AD[31:0] Valid Delay   | 2   | 11  | ns    | Min: 0pF<br>Max: 50pF | 15-3   |
| t41   | AD[31:0] Setup Time to PCICLK Rising   | 7   |     | ns    |                       | 15-4   |
| t42   | AD[31:0] Hold Time from PCICLK Rising  | 0   |     | ns    |                       | 15-4   |
| t43   | C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PAR, PERR#, PLOCK#, DEVSEL# Valid Delay from PCICLK Rising<br>Note: PERR# is for the ICH 82801AA Only.                | 2   | 11  | ns    | Min: 0pF<br>Max: 50pF | 15-3   |
| t44   | C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PAR, PERR#, PLOCK#, IDSEL, DEVSEL# Output Enable Delay from PCICLK Rising<br>Note: PERR# is for the ICH 82801AA Only. | 2   |     | ns    |                       | 15-7   |
| t45   | C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PERR#, PLOCK#, DEVSEL#, GNT[A:B]# Float Delay from PCICLK Rising<br>Note: PERR# is for the ICH 82801AA Only.          | 2   | 28  | ns    |                       | 15-5   |
| t46   | C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, SERR#, PERR#, DEVSEL#, Setup Time to PCICLK Rising<br>Note: PERR# is for the ICH 82801AA Only.                        | 7   |     | ns    |                       | 15-4   |
| t47   | C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, SERR#, PERR#, DEVSEL#, REQ[A:B]# Hold Time from PCLKIN Rising<br>Note: PERR# is for the ICH 82801AA Only.             | 0   |     | ns    |                       | 15-4   |
| t48   | PCIRST# Low Pulse Width  | 1   |     | ms    |                       | 15-6   |
| t49   | GNT[A:B]#, GNT[5:0]# Valid Delay from PCICLK Rising<br>Note: GNT[5:4]# is for the ICH 82801AA Only.  | 2   | 12  | ns    |                       |        |
| t50   | REQ[A:B]#, REQ[5:0]# Setup Timer to PCICLK Rising<br>Note: REQ[5:4]# is for the ICH 82801AA Only.  | 12  |     | ns    |                       |        |

Table 14-9. IDE PIO &amp; Multiword DMA Mode Timing

| Functional Operating Range (V <sub>REF</sub> = 5V ±5%, V <sub>CC</sub> = 3.3V ±0.3V, T <sub>CASE</sub> = 0°C to +100°C) |   |     |     |       |       |            |
|---|---|-----|-----|-------|-------|------------|
| Sym   | Parameter   | Min | Max | Units | Notes | Figure     |
| t60   | PDIOR#/PDIOW#/SDIOR#/SDIOW# Active From CLK66 Rising    | 2   | 20  | ns    |       | 15-8, 15-9 |
| t61   | PDIOR#/PDIOW#/SDIOR#/SDIOW# Inactive From CLK66 Rising  | 2   | 20  | ns    |       | 15-8, 15-9 |
| t62   | PDA[2:0]/SDA[2:0] Valid Delay From CLK66 Rising         | 2   | 30  | ns    |       | 15-8       |
| t63   | PDCS1#/SDCS1#, PDCS3#/SDCS3# Active From CLK66 Rising   | 2   | 30  | ns    |       | 15-8       |
| t64   | PDCS1#/SDCS1#, PDCS3#/SDCS3# Inactive From CLK66 Rising | 2   | 30  | ns    |       | 15-8       |
| t65   | PDDACK#/SDDACK# Active From CLK66 Rising                | 2   | 20  | ns    |       | 15-9       |
| t66   | PDDACK#/SDDACK# Inactive From CLK66 Rising              | 2   | 20  | ns    |       |            |
| t67   | PDDREQ/SDDREQ Setup Time to CLK66 Rising                | 7   |     | ns    |       | 15-9       |
| t68   | PDDREQ/SDDREQ Hold From CLK66 Rising                    | 7   |     | ns    |       | 15-9       |
| t69   | PDD[15:0]/SDD[15:0] Valid Delay From CLK66 Rising       | 2   | 30  | ns    |       | 15-8, 15-9 |
| t70   | PDD[15:0]/SDD[15:0] Setup Time to CLK66 Rising          | 10  |     | ns    |       | 15-8, 15-9 |
| t71   | PDD[15:0]/SDD[15:0] Hold From CLK66 Rising              | 7   |     | ns    |       | 15-8, 15-9 |
| t72   | PIORDY/SIORDY Setup Time to CLK66 Rising                | 7   |     | ns    | 1     | 15-8       |
| t73   | PIORDY/SIORDY Hold From CLK66 Rising                    | 7   |     | ns    | 1     | 15-8       |
| t74   | PIORDY/SIORDY Inactive Pulse Width                      | 48  |     | ns    |       | 15-8       |
| t75   | PDIOR#/PDIOW#/SDIOR#/SDIOW# Pulse Width Low             |     |     |       | 2,3   | 15-8, 15-9 |
| t76   | PDIOR#/PDIOW#/SDIOR#/SDIOW# Pulse Width High            |     |     |       | 3,4   | 15-8, 15-9 |

**NOTES:**

1. IORDY is internally synchronized. This timing is to guarantee recognition on the next clock.
2. PIORDY sample point from DIOx# assertion and PDIOx# active pulse width is programmable from 2-5 PCI clocks when the drive mode is Mode 2 or greater. Refer to the ISP field in the IDE Timing Register
3. PIORDY sample point from DIOx# assertion, PDIOx# active pulse width and PDIOx# inactive pulse width cycle time is the compatible timing when the drive mode is Mode 0/1. Refer to the TIM0/1 field in the IDE timing register.
4. PDIOx# inactive pulse width is programmable from 1-4 PCI clocks when the drive mode is Mode 2 or greater. Refer to the RCT field in the IDE Timing Register.

Table 14-10. Ultra DMA Timing (Mode 0, Mode 1, Mode 2)

| Functional Operating Range ( $V_{REF} = 5V \pm 5\%$ , $V_{CC} = 3.3V(0.3V)$ $T_{CASE} = 0^{\circ}C$ to $+100^{\circ}C$ ) |   |             |     |             |     |             |     |                 |
|--|---|-------------|-----|-------------|-----|-------------|-----|-----------------|
| Sym  | Parameter (1)                                 | Mode 0 (ns) |     | Mode 1 (ns) |     | Mode 2 (ns) |     | Figure          |
|  |   | Min         | Max | Min         | Max | Min         | Max |                 |
| t80  | Sustained Cycle Time (T2cyc <sub>typ</sub> )  | 240         |     | 160         |     | 120         |     |                 |
| t81  | Cycle Time (T <sub>cyc</sub> )                | 112         |     | 73          |     | 54          |     | 15-11           |
| t82  | Two Cycle Time (T2cyc)                        | 230         |     | 154         |     | 115         |     | 15-11           |
| t83  | Data Setup Time (T <sub>ds</sub> )            | 15          |     | 10          |     | 7           |     | 15-11           |
| t84  | Data Hold Time (T <sub>dh</sub> )             | 5           |     | 5           |     | 5           |     | 15-11           |
| t85  | Data Valid Setup Time (T <sub>dvs</sub> )     | 70          |     | 48          |     | 30          |     | 15-11           |
| t86  | Data Valid Hold Time (T <sub>dvh</sub> )      | 6           |     | 6           |     | 6           |     | 15-11           |
| t87  | Limited Interlock Time (T <sub>li</sub> )     | 0           | 150 | 0           | 150 | 0           | 150 | 15-13           |
| t88  | Interlock Time w/ Minimum (T <sub>mli</sub> ) | 20          |     | 20          |     | 20          |     | 15-13           |
| t89  | Envelope Time (T <sub>env</sub> )             | 20          | 70  | 20          | 70  | 20          | 70  | 15-10           |
| t90  | Ready to Pause Time (T <sub>rp</sub> )        | 160         |     | 125         |     | 100         |     | 15-12           |
| t91  | DMACK setup/hold Time (T <sub>ack</sub> )     | 20          |     | 20          |     | 20          |     | 15-10,<br>15-13 |

**NOTE:**

1. The specification symbols in parentheses correspond to the Ultra DMA specification name.

Table 14-11. Ultra DMA Timing (Mode 3, Mode 4) (ICH 82801AA Only)

| Functional Operating Range ( $V_{REF} = 5V \pm 5\%$ , $V_{CC} = 3.3V(0.3V)$ $T_{CASE} = 0^{\circ}C$ to $+100^{\circ}C$ ) |   |             |     |             |     |                 |  |  |
|--|---|-------------|-----|-------------|-----|-----------------|--|--|
| Sym  | Parameter (1)                                 | Mode 3 (ns) |     | Mode 4 (ns) |     | Figure          |  |  |
|  |   | Min         | Max | Min         | Max |                 |  |  |
| t80  | Sustained Cycle Time (T2cyc <sub>typ</sub> )  | 90          |     | 60          |     |                 |  |  |
| t81  | Cycle Time (T <sub>cyc</sub> ) (2)            | 39          |     | 25          |     | 15-11           |  |  |
| t82  | Two Cycle Time (T2cyc)                        | 86          |     | 57          |     | 15-11           |  |  |
| t83  | Data Setup Time (T <sub>ds</sub> )            | 7           |     | 5           |     | 15-11           |  |  |
| t84  | Data Hold Time (T <sub>dh</sub> )             | 5           |     | 5           |     | 15-11           |  |  |
| t85  | Data Valid Setup Time (T <sub>dvs</sub> )     | 20          |     | 6           |     | 15-11           |  |  |
| t86  | Data Valid Hold Time (T <sub>dvh</sub> )      | 6           |     | 6           |     | 15-11           |  |  |
| t87  | Limited Interlock Time (T <sub>li</sub> )     | 0           | 100 | 0           | 100 | 15-13           |  |  |
| t88  | Interlock Time w/ Minimum (T <sub>mli</sub> ) | 20          |     | 20          |     | 15-13           |  |  |
| t89  | Envelope Time (T <sub>env</sub> )             | 20          | 55  | 20          | 55  | 15-10           |  |  |
| t90  | Ready to Pause Time (T <sub>rp</sub> )        | 100         |     | 100         |     | 15-12           |  |  |
| t91  | DMACK setup/hold Time (T <sub>ack</sub> )     | 20          |     | 20          |     | 15-10,<br>15-13 |  |  |

Table 14-12. Universal Serial Bus Timing

| Functional Operating Range ( $V_{REF} = 5V \pm 5\%$ , $V_{CC} = 3.3V \pm 0.3V$ , $T_{CASE} = 0^{\circ}C$ to $+100^{\circ}C$ ) |  |            |          |          |   |       |
|---|--|------------|----------|----------|---|-------|
| Sym   | Parameter                                    | Min        | Max      | Units    | Notes                                   | Fig   |
| Full Speed Source (Note 7)  |  |            |          |          |   |       |
| t100  | USBPx+, USBPx- Driver Rise Time              | 4          | 20       | ns       | 1, $C_L = 50$ pF                        | 15-14 |
| t101  | USBPx+, USBPx- Driver Fall Time              | 4          | 20       | ns       | 1, $C_L = 50$ pF                        | 15-14 |
| t102  | Source Differential Driver Jitter            |            |          |          |   |       |
|   | To Next Transition<br>For Paired Transitions | -2<br>-1   | 2<br>1   | ns<br>ns | 2, 3                                    | 15-15 |
| t103  | Source EOP Width                             | 160        | 175      | ns       | 4                                       | 15-16 |
| t104  | Differential to SE0 Transition Skew          | -2         | 5        | ns       | 5                                       |       |
| t105  | Receiver Data Jitter Tolerance               |            |          |          |   |       |
|   | To Next Transition<br>For Paired Transitions | -20<br>-10 | 20<br>10 | ns<br>ns | 3                                       | 15-15 |
| t106  | EOP Width: Must reject as EOP                | 40         |          | ns       | 4                                       | 15-16 |
|   | EOP Width: Must accept as EOP                | 85         |          | ns       |   |       |
| t107  | Differential to SE0 Transition Skew          | -2         | 5        | ns       | 5                                       |       |
| Low Speed Source (Note 8)   |  |            |          |          |   |       |
| t108  | USBPx+, USBPx- Driver Rise Time              | 75         | 300      | ns       | 1, 6<br>$C_L = 50$ pF<br>$C_L = 350$ pF | 15-14 |
|   |  |            |          | ns       |   |       |
| t109  | USBPx+, USBPx- Driver Fall Time              | 75         | 300      | ns       | 1, 6<br>$C_L = 50$ pF<br>$C_L = 350$ pF | 15-14 |
|   |  |            |          | ns       |   |       |
| t110  | Source Differential Driver Jitter            |            |          |          |   |       |
|   | To Next Transition<br>For Paired Transitions | -2<br>-1   | 2<br>1   | ns<br>ns | 2, 3                                    | 15-15 |
| t111  | Source EOP Width                             | 160        | 175      | ns       | 4                                       | 15-16 |
| t112  | Differential to SE0 Transition Skew          | -2         | 5        | ns       | 5                                       |       |
| t113  | Receiver Data Jitter Tolerance               |            |          |          |   |       |
|   | To Next Transition<br>For Paired Transitions | -20<br>-10 | 20<br>10 | ns<br>ns | 3                                       | 15-15 |
| t114  | EOP Width: Must reject as EOP                | 40         |          | ns       | 4                                       | 15-16 |
|   | EOP Width: Must accept as EOP                | 85         |          | ns       |   |       |
| t115  | Differential to SE0 Transition Skew          | -2         | 5        | ns       | 5                                       |       |

**NOTES:**

1. Driver output resistance under steady state drive is specified at 28 ohms at minimum and 43 ohms at maximum
2. Timing difference between the differential data signals
3. Measured at crossover point of differential data signals
4. Measured at 50% swing point of data signals
5. Measured from last crossover point to 50% swing point of data line at leading edge of EOP
6. Measured from 10% to 90% of the data signal
7. Full Speed Data Rate has minimum of 11.97 Mbps and maximum of 12.03 Mbps
8. Low Speed Data Rate has a minimum of 1.48 Mbps and a maximum of 1.52 Mbps

Table 14-13. IOAPIC Bus Timing

| Functional Operating Range ( $V_{REF} = 5V \pm 5\%$ , $V_{CC} = 3.3V \pm 0.3V$ $T_{CASE} = 0^{\circ}C$ to $+100^{\circ}C$ ) |  |     |      |       |       |      |
|---|--|-----|------|-------|-------|------|
| Sym   | Parameter                                    | Min | Max  | Units | Notes | Fig  |
| t120  | APICCD[1:0]# Valid Delay from APICCLK Rising | 3.0 | 12.0 | ns    |       | 15-3 |
| t121  | APICCD[1:0]# Setup Time to APICCLK Rising    | 8.5 |      | ns    |       | 15-4 |
| t122  | APICCD[1:0]# Hold Time from APICCLK Rising   | 3.0 |      | ns    |       | 15-4 |

Table 14-14. SMBus Timing

| Functional Operating Range ( $V_{REF} = 5V \pm 5\%$ , $V_{CC} = 3.3V \pm 0.3V$ $T_{CASE} = 0^{\circ}C$ to $+100^{\circ}C$ ) |  |     |     |       |       |       |
|---|--|-----|-----|-------|-------|-------|
| Sym   | Parameter  | Min | Max | Units | Notes | Fig   |
| t130  | Bus Free Time Between Stop and Start Condition   | 4.7 |     | us    |       | 15-17 |
| t131  | Hold Time after (repeated) Start Condition. After this period, the first clock is generated. | 4.0 |     | us    |       | 15-17 |
| t132  | Repeated Start Condition Setup Time  | 4.7 |     | us    |       | 15-17 |
| t133  | Stop Condition Setup Time  | 4.0 |     | us    |       | 15-17 |
| t134  | Data Hold Time   | 300 |     | ns    |       | 15-17 |
| t135  | Data Setup Time  | 250 |     | ns    |       | 15-17 |
| t136  | Device Time Out  | 25  | 35  | ms    | 1     |       |
| t137  | Cumulative Clock Low Extend Time (slave device)  |     | 25  | ms    | 2     | 15-18 |
| t138  | Cumulative Clock Low Extend Time (master device)   |     | 10  | ms    | 3     | 15-18 |

**NOTES:**

1. A device will timeout when any clock low exceeds this value.
2. t137 is the cumulative time a slave device is allowed to extend the clock cycles in one message from the initial start to stop. If a slave device exceeds this time, it is expected to release both its clock and data lines and reset itself.
3. t138 is the cumulative time a master device is allowed to extend its clock cycles within each byte of a message as defined from start-to-ack, ack-to-ack or ack-to-stop.

Table 14-15. AC'97 Timing

| Functional Operating Range ( $V_{REF} = 5V \pm 5\%$ , $V_{CC} = 3.3V \pm 0.3V$ $T_{CASE} = 0^{\circ}C$ to $+100^{\circ}C$ ) |  |     |     |       |       |      |
|---|--|-----|-----|-------|-------|------|
| Sym   | Parameter  | Min | Max | Units | Notes | Fig  |
| t140  | ACSDIN[0:1] Setup to Falling Edge of BITCLK            | 15  |     | ns    |       |      |
| t141  | ACSDIN[0:1] Hold from Falling Edge of BITCLK           | 5   |     | ns    |       |      |
| t142  | ACSYNC, ACSDOUT valid delay from rising edge of BITCLK |     | 15  | ns    |       | 15-3 |



Table 14-16. LPC Timing

| Functional Operating Range (VREF = 5V ±5%, Vcc=3.3v±0.3V TCASE = 0°C to +100°C) |   |     |     |       |       |      |
|---|---|-----|-----|-------|-------|------|
| Sym   | Parameter                                       | Min | Max | Units | Notes | Fig  |
| t150  | LAD[3:0] Valid Delay from PCICLK Rising         | 2   | 11  | ns    |       | 15-3 |
| t151  | LAD[3:0] Output Enable Delay from PCICLK Rising | 2   |     | ns    |       | 15-7 |
| t152  | LAD[3:0] Float Delay from PCICLK Rising         |     | 28  | ns    |       | 15-5 |
| t153  | LAD[3:0] Setup Time to PCICLK Rising            | 7   |     | ns    |       | 15-4 |
| t154  | LAD[3:0] Hold Time from PCICLK Rising           | 0   |     | ns    |       | 15-4 |
| t155  | LDRQ[1:0]# Setup Time to PCICLK Rising          | 12  |     | ns    |       | 15-4 |
| t156  | LDRQ[1:0]# Hold Time from PCICLK Rising         | 0   |     | ns    |       | 15-4 |
| t157  | LFRAME# Valid Delay from PCICLK Rising          | 2   | 12  | ns    |       | 15-3 |

Table 14-17. Miscellaneous Timings

| Functional Operating Range (VREF = 5V ±5%, Vcc=3.3v±0.3V TCASE = 0°C to +100°C) |   |     |     |         |       |      |
|---|---|-----|-----|---------|-------|------|
| Sym   | Parameter                                 | Min | Max | Units   | Notes | Fig  |
| t160  | SERIRQ Setup Time to PCICLK Rising        | 7   |     | ns      |       | 15-4 |
| t161  | SERIRQ Hold Time from PCICLK Rising       | 0   |     | ns      |       | 15-4 |
| t162  | RI#, EXTSMI#, GPI, USB Resume Pulse Width | 2   |     | RTCCCLK |       | 15-6 |
| t163  | SPKR Valid Delay from OSC Rising          |     | 200 | ns      |       | 15-3 |
| t164  | SERR# Active to NMI Active                |     | 200 | ns      |       |      |
| t165  | IGNNE# Inactive from FERR# Inactive       |     | 230 | ns      |       |      |

Table 14-18. Reset Timing

| Functional Operating Range (VREF = 5V ±5%, Vcc=3.3v±0.3V TCASE = 0°C to +100°C) |  |       |     |         |       |     |
|---|--|-------|-----|---------|-------|-----|
| Sym   | Parameter  | Min   | Max | Units   | Notes | Fig |
| t170  | PCIRST# Driven Inactive After SUS_STAT# is Driven Inactive |       | 1   | RTCCCLK |       |     |
| t171  | PCIRST# Active Pulse Width (via RC Register)               | 1     |     | ms      |       |     |
| t172  | PWROK Rise Time  |       | 10  | ns      |       |     |
| t173  | PWROK Active to PCIRST# Inactive                           | 1     |     | ms      |       |     |
| t175  | AC_RST# Active Low Pulse Width                             | 1     |     | us      |       |     |
| t176  | AC_RST# Inactive to BIT_CLK Startup Delay                  | 162.8 |     | ns      |       |     |
| t177  | VccRTC Good to RTCRST# Inactive                            | 10    | 20  | ms      |       |     |

Table 14-19. Power Management Timing (Sheet 1 of 2)

| Functional Operating Range ( $V_{REF} = 5V \pm 5\%$ , $V_{CC} = 3.3V \pm 0.3V$ , $T_{CASE} = 0^{\circ}C$ to $+100^{\circ}C$ ) |   |     |      |        |       |                         |
|---|---|-----|------|--------|-------|-------------------------|
| Sym   | Parameter   | Min | Max  | Units  | Notes | Fig                     |
| t180  | VccSUS Good to SLP_S3#, SLP_S5#, and PCIRST# Active           |     | 50   | ns     |       | 15-19                   |
| t181  | VccSUS Good to RSMRST# Inactive                               | 1   |      | ms     | 6     | 15-19                   |
| t182  | RSMRST# Inactive to SLP_S3#, SLP_S5#, SUS_STAT# Inactive      | 1   | 4    | RTCCLK |       | 15-19                   |
| t183  | VccSUS Good to Vcc3_3 (core) Good                             | 0   |      | ns     | 4     | 15-19                   |
| t184  | STPCLK# Active to Stop Grant Acknowledge Cycle                | N/A | N/A  |        | 2     | 15-20<br>15-21<br>15-22 |
| t185  | Stop Grant Acknowledge Cycle to CPUSLP# Active                | 8   |      | PCICLK | 4     | 15-20<br>15-21<br>15-22 |
| t186  | CPUSLP# Active to SUS_STAT# Active                            |     | 1    | RTCCLK | 1     | 15-21<br>15-22          |
| t187  | SUS_STAT# Active to PCIRST# Active and CPUSLP# Invalid        | 2   | 3    | RTCCLK | 1     | 15-21<br>15-22          |
| t188  | PCIRST# Active to SLP_S3# Active                              | 1   | 2    | RTCCLK | 1     | 15-21<br>15-22          |
| t189  | SLP_S3# Active to SLP_S5# Active                              | 1   | 2    | RTCCLK | 1, 7  | 15-22                   |
| t190  | SLP_S3# or SLP_S5# Active to PWROK Inactive                   | 0   |      | ns     | 4, 5  | 15-21<br>15-22          |
| t191  | PWROK Inactive to Vcc3_3(core) Not Good                       | 20  |      | ns     | 6     | 15-21<br>15-22          |
| t192  | Wake Event to SLP_S3#, SLP_S5# Inactive                       | 2   | 3    | RTCCLK | 1     | 15-21<br>15-22          |
| t193  | SLP_S3#, SLP_S5# Inactive to Vcc3_3(core) Good                | 0   |      | ns     | 4     | 15-19<br>15-21<br>15-22 |
| t194  | Vcc3_3(core) Good to STPCLK# Active, CPUSLP# Invalid          |     | 50   | ns     |       | 15-19<br>15-21<br>15-22 |
| t195  | Vcc3_3 (core) Good to PWROK Active                            | 1   |      | ms     | 6     | 15-19<br>15-21<br>15-22 |
| t196  | PWROK Active to PCIRST# Inactive                              | 1.0 | 1.16 | ms     |       | 15-19<br>15-21<br>15-22 |
| t197  | PCIRST# Inactive to STPCLK# Inactive                          | 1   | 4    | PCICLK |       | 15-19<br>15-21<br>15-22 |
| t198  | SUS_STAT# Inactive to PCIRST# Inactive                        | 2   |      | RTCCLK | 1     | 15-21<br>15-22          |
| t199  | CPUSLP# Inactive to STPCLK# Inactive                          | 32  | 36   | PCICLK |       | 15-20                   |
| t200  | Wake Event from S1 to CPUSLP# Inactive                        |     | 4    | PCICLK |       | 15-20                   |
| t201  | For the ICH (82801AA), RSMRST# inactive to SUSCLK Oscillating | 1   | 2    | RTCCLK |       | 15-19                   |

Table 14-19. Power Management Timing (Sheet 2 of 2)

| Functional Operating Range (VREF = 5V ±5%, Vcc=3.3v±0.3V TCASE = 0°C to +100°C) |                               |     |     |       |       |     |
|---|-------------------------------|-----|-----|-------|-------|-----|
| Sym   | Parameter                     | Min | Max | Units | Notes | Fig |
| t203  | VccSUS Removed to RSMRST# Low |     | 25  | ms    |       |     |

**NOTES:**

1. These transitions are clocked off the internal RTC. 1 RTC clock is approximately 32us.
2. The ICH STPCLK# assertion will trigger the processor to send a stop grant acknowledge cycle. The timing for this cycle getting to the ICH is dependant on the processor and the memory controller.
3. The ICH will wait for the processor's stop grant acknowledge cycle before asserting processorSLP#. Asserting CPUSLP# is optional and is enabled by setting the ICH CPUSLP\_EN bit in the General PM Configuration 1 Register.
4. The ICH has no timing requirement for this transition. This transition depends on external implementation.
5. It is up to the system designer to determine if the SLP\_S3# and SLP\_S5# signals are used to control the power planes.
6. This specification must be guaranteed by external design.
7. If the transition to S5 is due to Power Button Override, SLP\_S3# and SLP\_S5# are asserted together following timing t188 (PCIRST# active to SLP\_S3# and SLP\_S5# active).

## 14.5 Timing Diagrams

Figure 14-1. Clock Timing

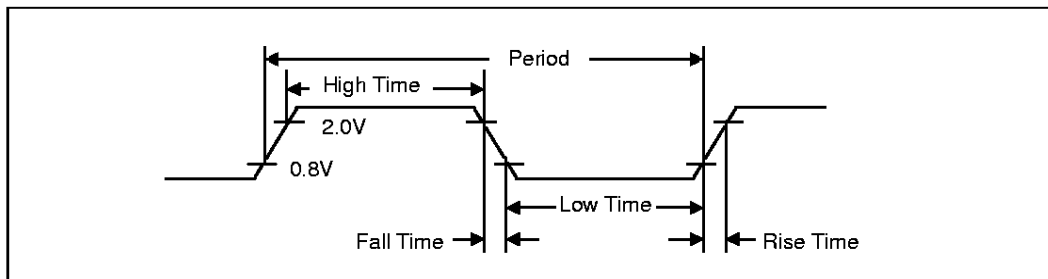


Figure 14-2. Valid Delay From Rising Clock Edge

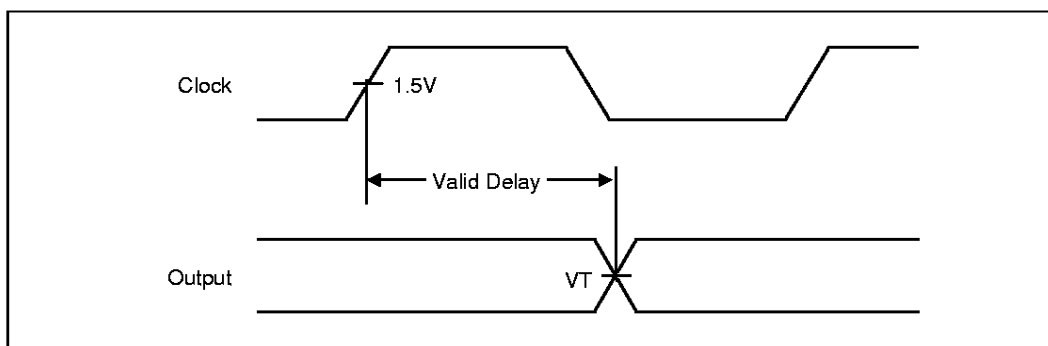


Figure 14-3. Setup And Hold Times

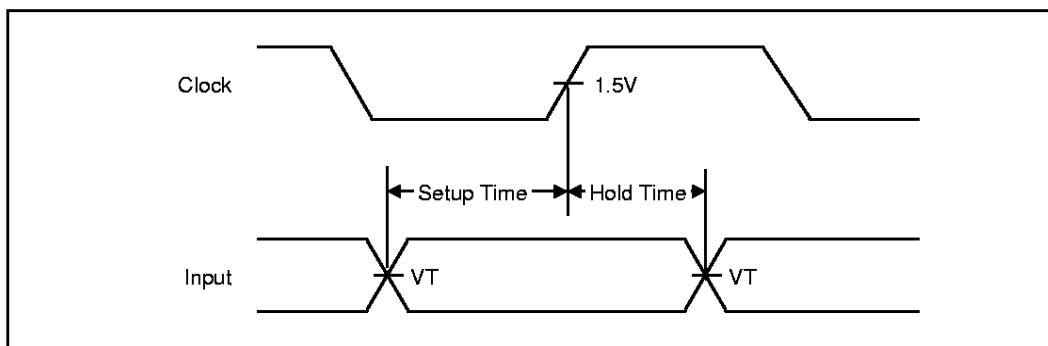


Figure 14-4. Float Delay

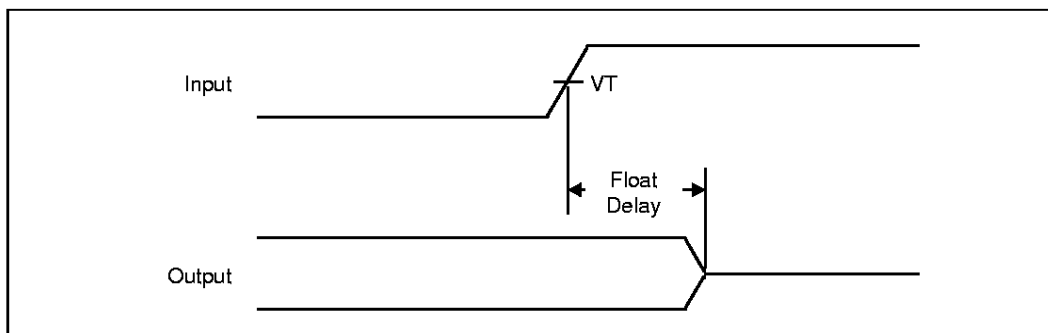


Figure 14-5. Pulse Width

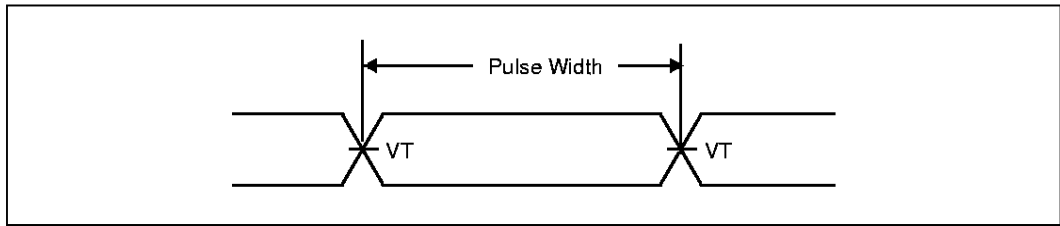


Figure 14-6. Output Enable Delay

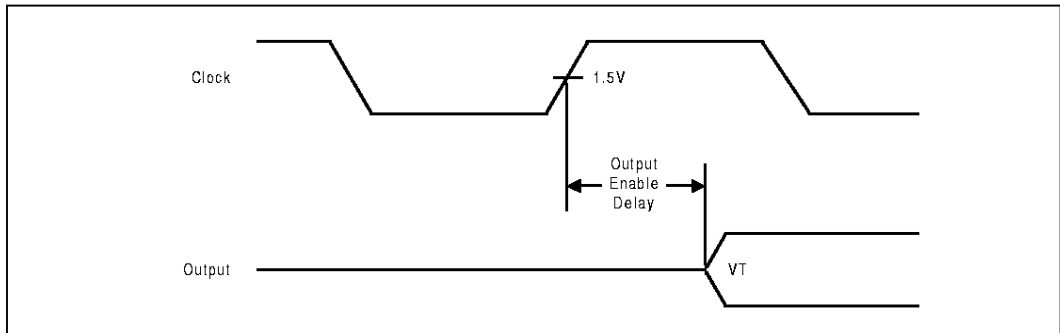


Figure 14-7. IDE PIO Mode

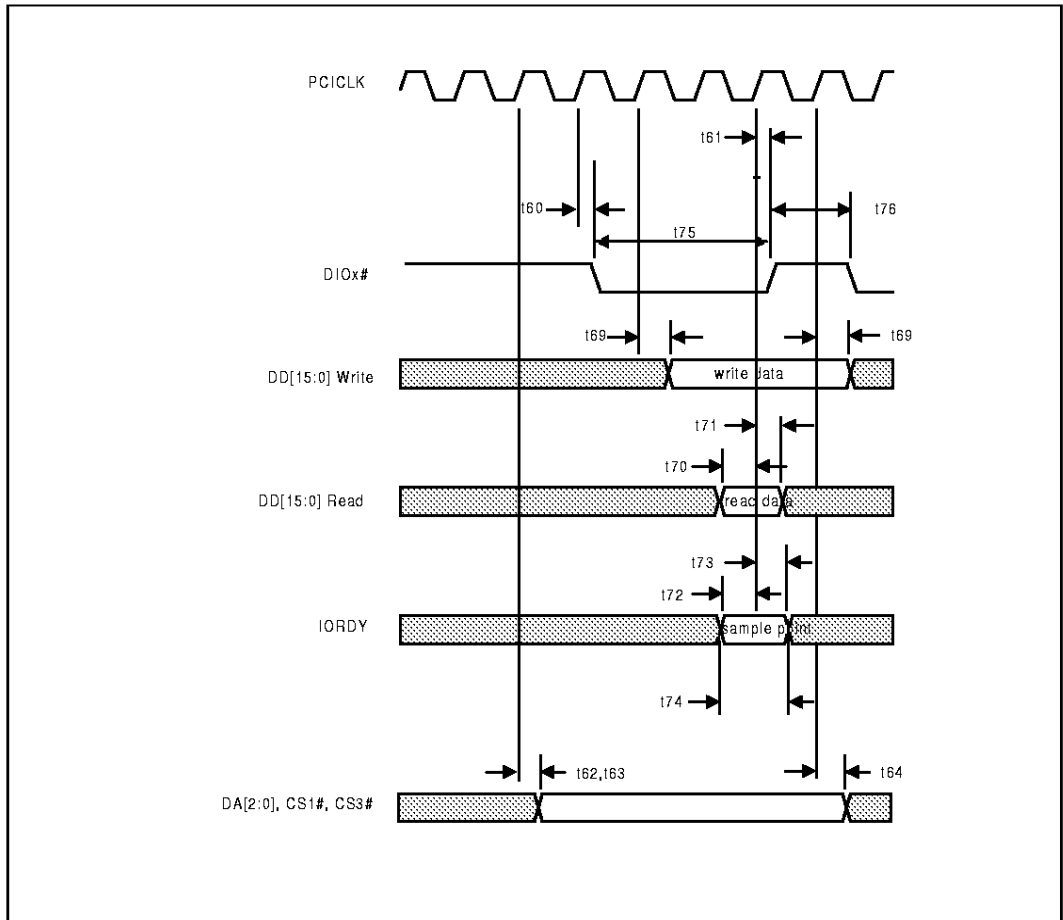


Figure 14-8. IDE Multiword DMA

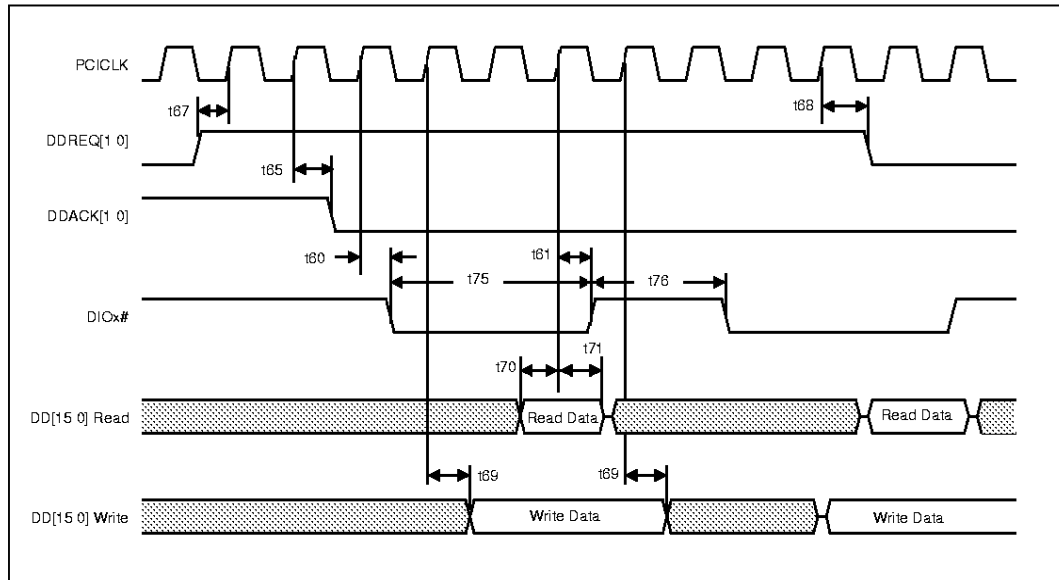


Figure 14-9. Ultra DMA Mode (Drive Initiating a Burst Read)

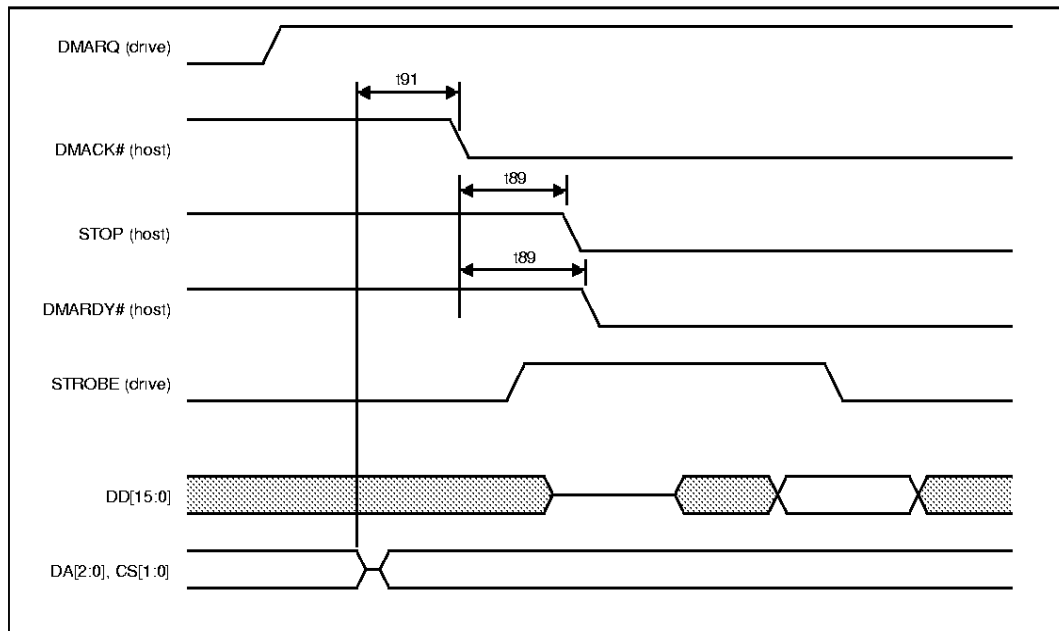


Figure 14-10. Ultra DMA Mode (Sustained Burst)

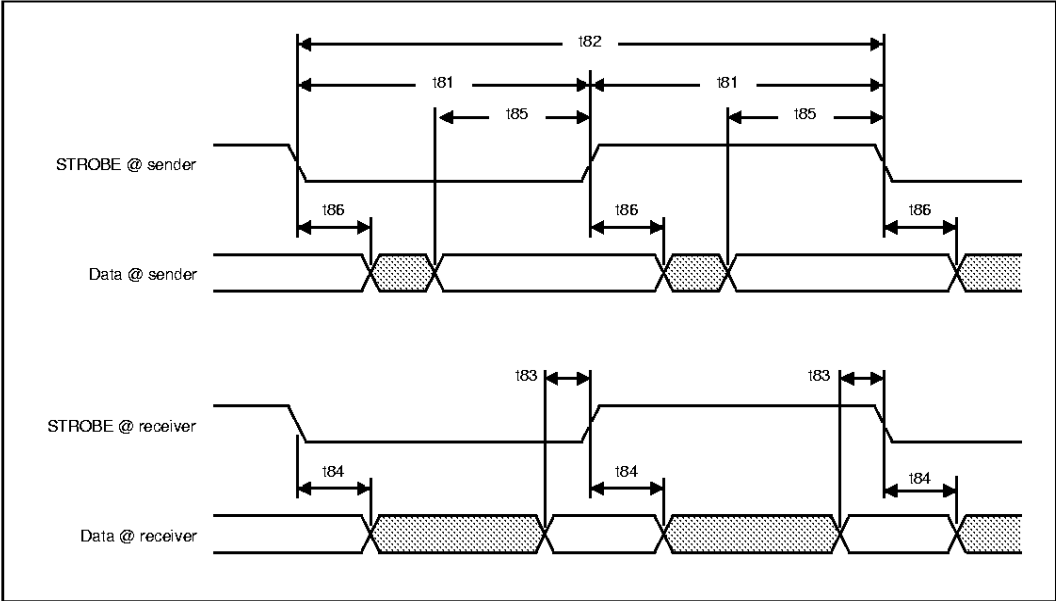


Figure 14-11. Ultra DMA Mode (Pausing a DMA Burst)

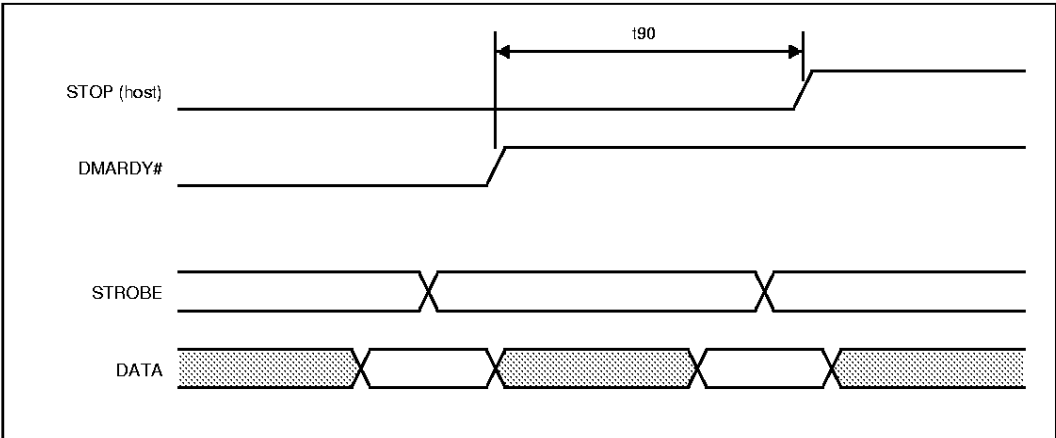




Figure 14-12. Ultra DMA Mode (Terminating a DMA Burst)

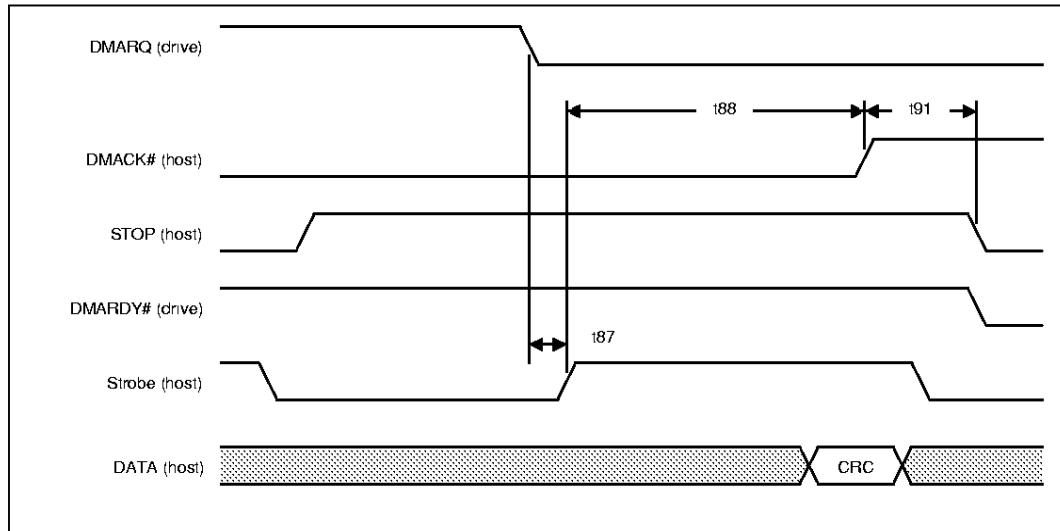


Figure 14-13. USB Rise and Fall Times

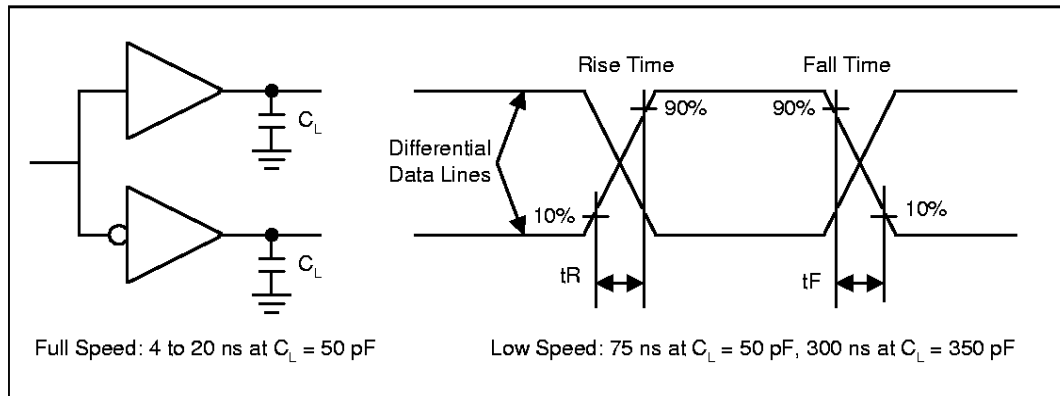


Figure 14-14. USB Jitter

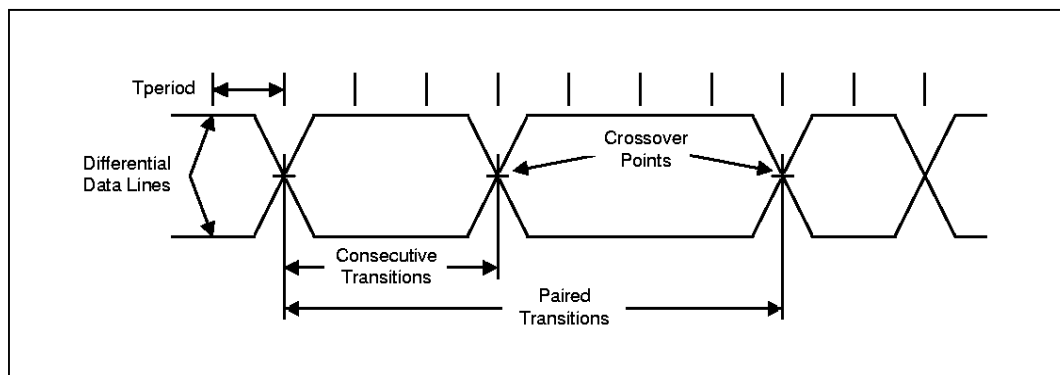


Figure 14-15. USB EOP Width

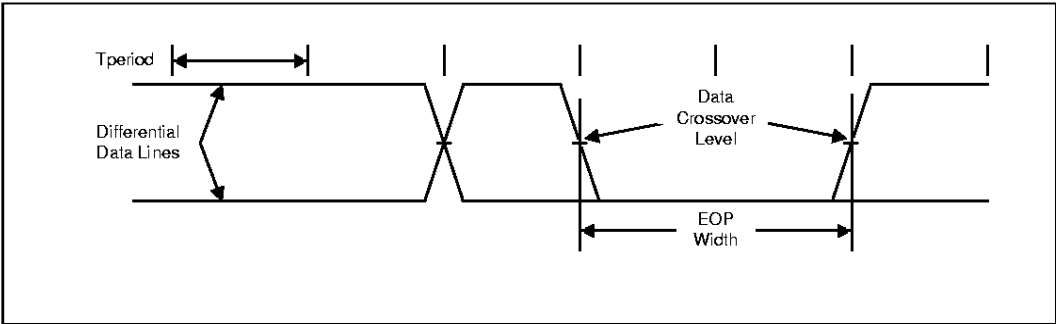


Figure 14-16. SMBus Transaction

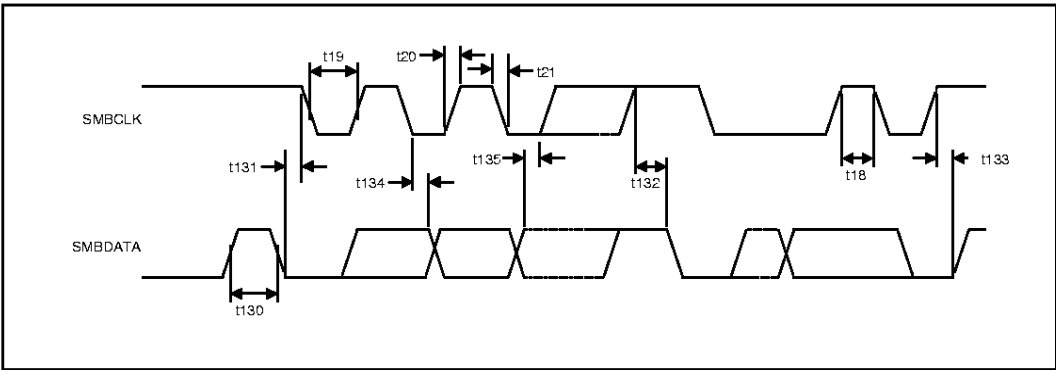


Figure 14-17. SMBus Timeout

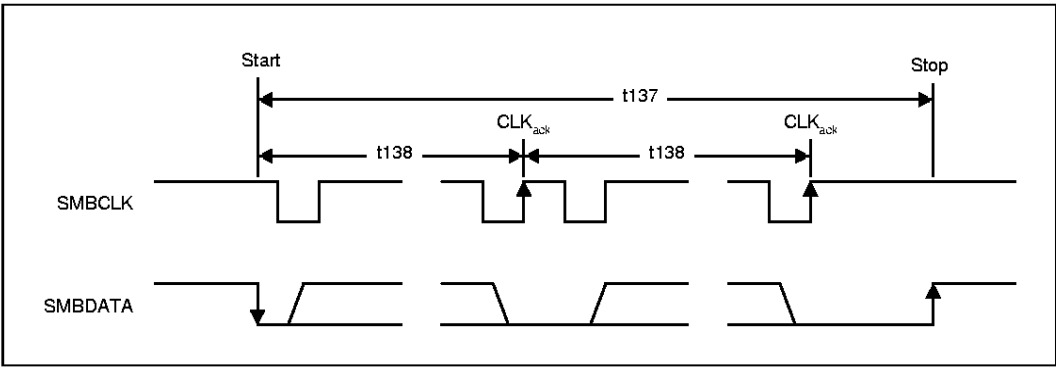


Figure 14-18. G3 (Mechanical Off) to S0 Timings

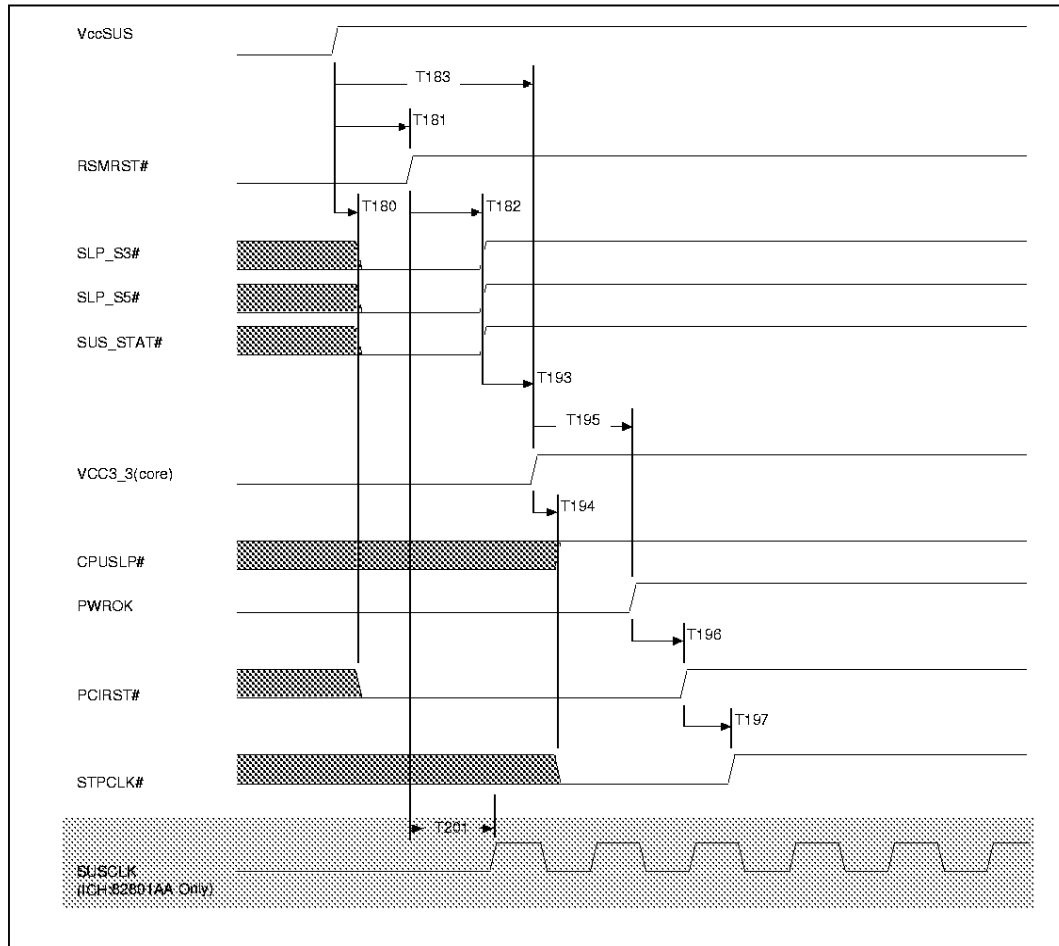


Figure 14-19. S0 to S1 to S0 Timings

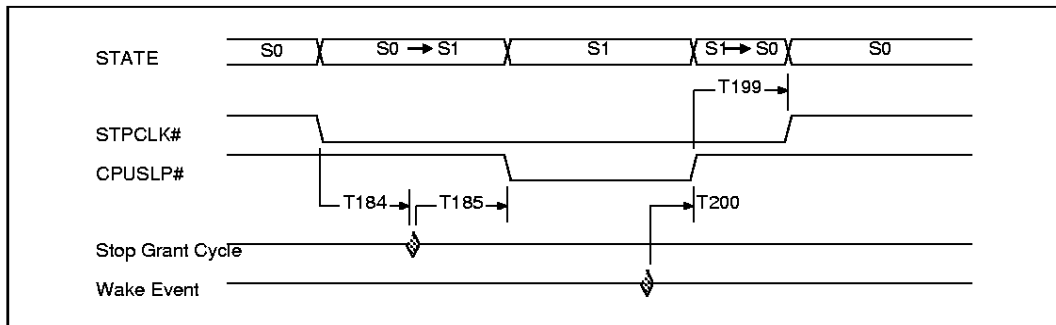


Figure 14-20. S0 to S3 to S0 Timings

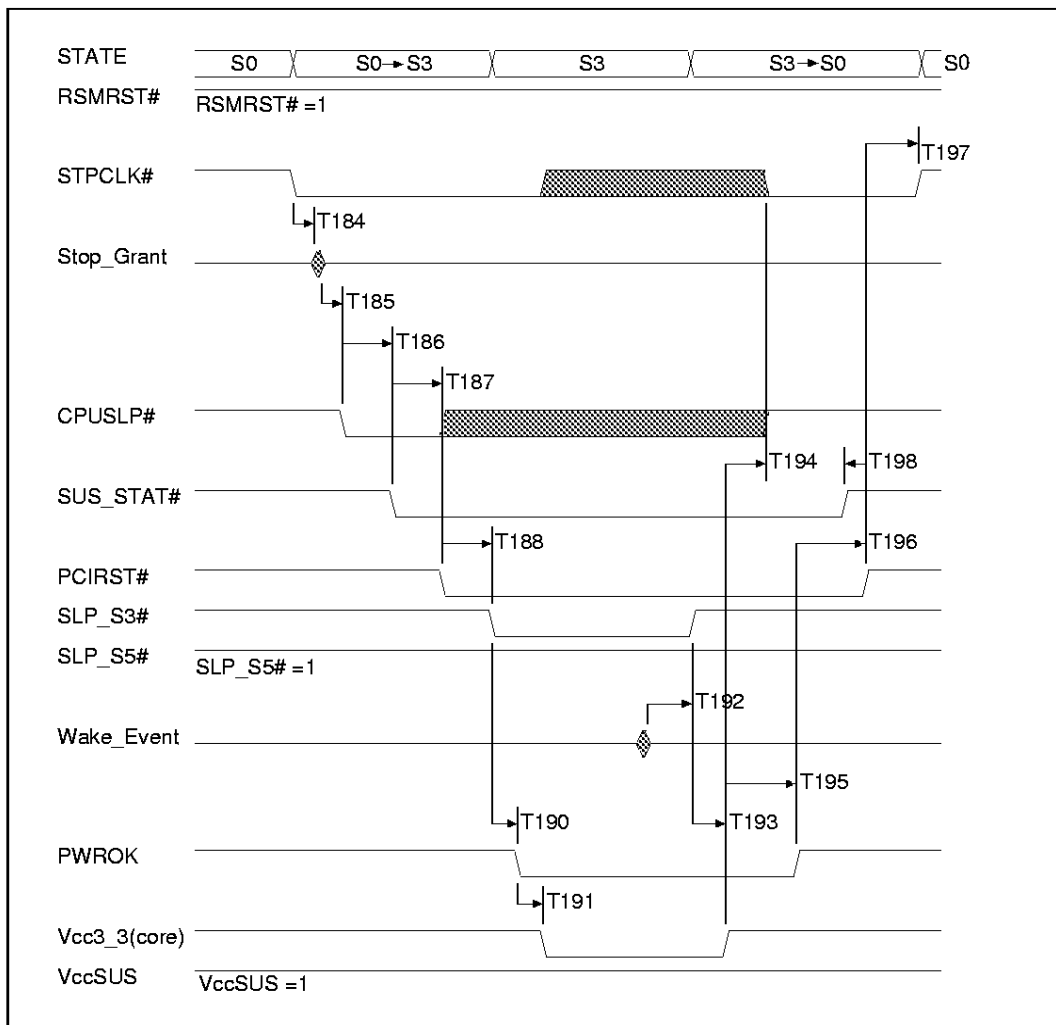
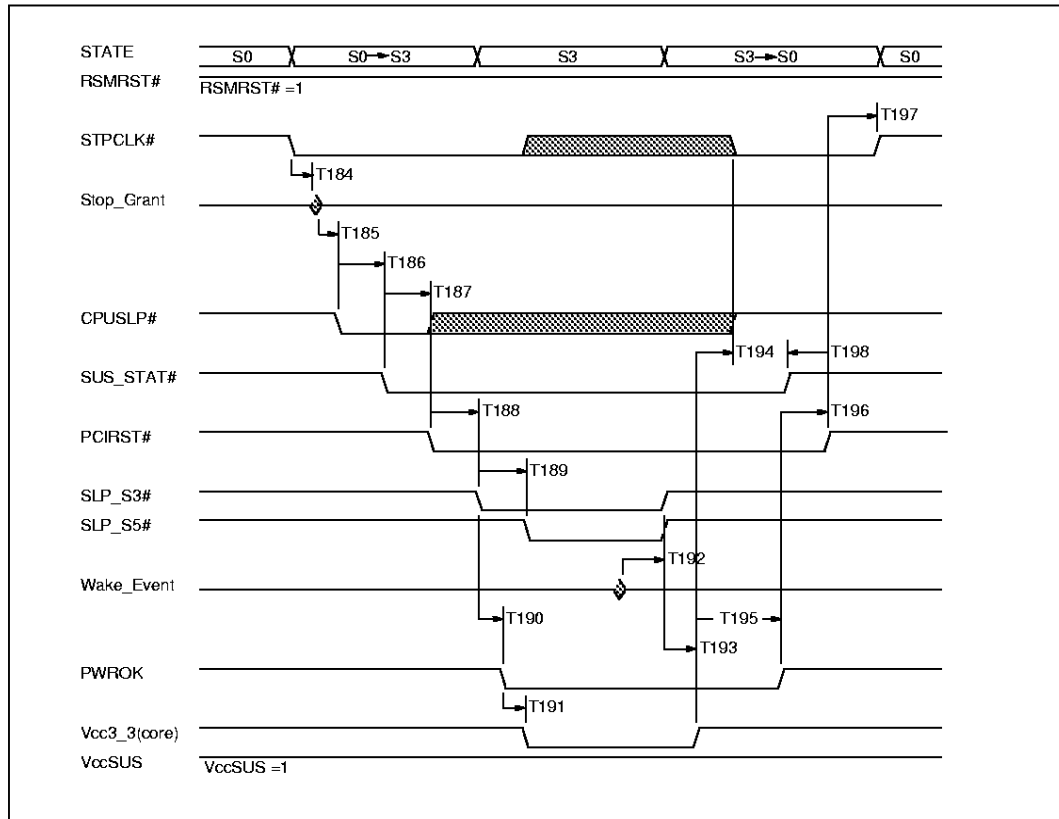


Figure 14-21. S0 to S5 to S0 Timings





# ***Pinout and Package Information***

---

# **15**

## **15.1 Pinout Information**

Figure 15-1, Figure 15-2, and Table 15-1 show the ballout for both the ICH (82801AA) and ICH0 (82801AB).

Figure 15-1. ICH (82801AA) and ICH0 (82801AB) Ballout—Top View

|   | 1                      | 2        | 3                    | 4  | 5   | 6                 | 7      | 8      | 9                               |     |     |
|---|------------------------|----------|----------------------|--|---|-------------------|--------|--------|---------------------------------|-----|-----|
| A | SERR#                  | IRDY#    | C/BE2#               | AD16   | VCC3_3  | AD22              | AD26   | AD27   | PAR                             |     |     |
| B | AD13                   | C/BE1#   | FRAME#               | AD17   | AD20  | AD23              | AD28   | AD25   | PLOCK#                          |     |     |
| C | AD12                   | AD11     | AD15                 | TRDY#  | AD18  | AD19              | AD31   | VCC3_3 | AD29                            |     |     |
| D | AD8                    | C/BE0#   | AD9                  | AD14   | STOP#   | C/BE3#            | AD24   | AD30   | DEVSEL#                         |     |     |
| E | AD6                    | AD7      | VCC3_3               | AD10   | VCC3_3  | VCC3_3            | AD21   |        | PERR# <sup>1</sup> /<br>GPIO[7] |     |     |
| F | RSMRST#                | AD2      | AD3                  | AD4  | AD5   |                   |        |        |                                 |     |     |
| G | VCCRTC                 | AD0      | VSS                  | AD1  | VCC3_3  |                   |        |        |                                 |     |     |
| H | RTCRST#                | VBIAS    | RTCX1                | RTCX2  |   |                   |        |        | VSS                             | VSS |     |
| J | SMBDATA                | SMBCLK   | PWROK                | GPIO[10]/<br>INTRUDER#                       | PCIRST#                                       |                   |        |        |                                 | VSS | VSS |
| K | PME#                   | SLP_S5#  | GPIO[24]/<br>SLP_S3# | SUSCLK <sup>1</sup> /<br>GPIO[26]            |   |                   |        |        | VSS                             | VSS |     |
| L | VCCSUS                 | GPIO[13] | RI#                  | GPIO[25]/<br>SUSSTAT#                        | GPIO[28]/<br>ALERTDATA <sup>1</sup>           |                   |        |        |                                 |     |     |
| M | GPIO[11]/<br>SMBALERT# | PWRBTN#  | OC0#                 | OC1#   | GPIO[27]/<br>ALERTCLK <sup>1</sup>            |                   |        |        |                                 |     |     |
| N | VCCSUS                 | USBP0N   | GPIO[8]/<br>LDRQ1#   | GPIO[12]                                     | VCC3_3  | GPIO[0]/<br>REQA# | PDD9   |        | PDD1                            |     |     |
| P | USBP0P                 | USBP1N   | GPIO[9]/<br>AC_SDIN1 | GPIO[1]/<br>REQ[B]#/<br>REQ[5]# <sup>1</sup> | GPIO[16]/<br>GNT[A]#                          | VCC3_3            | PDD8   | PDD11  | PDD13                           |     |     |
| R | USBP1P                 | VSS      | AC_BITCLK            | SERIRQ                                       | GPIO[17]/<br>GNT[B]#/<br>GNT[5]# <sup>1</sup> | LAD0/ FWH0        | PDD6   | PDD4   | PDD2                            |     |     |
| T | AC_RST#                | AC_SDOUT | AC_SYNC              | LAD3/ FWH3                                   | LAD2/ FWH2                                    | LDRQ0#            | VCC3_3 | PDD10  | PDD12                           |     |     |
| U | AC_SDIN0               | CLK48    | SPKR                 | LFRAME#/<br>FWH4                             | LAD1/ FWH1                                    | CLK14             | PDD7   | PDD5   | PDD3                            |     |     |

**NOTES:**

1. PERR#, REQ4#, REQ5#, GNT4#, GNT5#, SUSCLK, HL[11], ALERTCLK, and ALERTDATA are on the ICH (82801AA only). These signals are not on the ICH0 (82801AB).

The non-multiplexed signals in this group (REQ4#, GNT4, and HL[11]) are Reserved on ICH0 (No Connect).



Figure 15-2. ICH (82801AA) and ICH0 (82801AB) Ballout—Top View

|        | 10     | 11                   | 12      | 13       | 14       | 15       | 16                | 17      |   |
|--------|--------|----------------------|---------|----------|----------|----------|-------------------|---------|---|
|        | PIRQB# | GNT[4]# <sup>1</sup> | GNT[2]# | GNT[0]#  | REQ[0]#  | RCIN#    | CLK66             | STPCLK# | A |
|        | PIRQC# | REQ[4]# <sup>1</sup> | REQ[2]# | REQ[1]#  | GPIO[21] | A20GATE  | NMI               | IGNNE#  | B |
|        | PIRQD# | VCC3_3               | GNT[3]# | GNT[1]#  | PCICLK   | 5VREF    | APICCLK           | APICD1  | C |
|        | PIRQA# | GPIO[5]              | REQ[3]# | GPIO[22] | THRM#    | GPIO[23] | VCC3_3            | HL0     | D |
|        |        | GPIO[6]              | CPUSLP# | VCC3_3   | INTR     | INIT#    | APICD0            | HL1     | E |
|        |        |                      |         | A20M#    | SMI#     | FERR#    | HL11 <sup>1</sup> | HL2     | F |
|        |        |                      |         | VCC1_8   | VSS      | VCC1_8   | HL3               | HL_STB  | G |
| VSS    |        |                      |         |          | VCC1_8   | HL8      | VCC1_8            | HL_STB# | H |
| VSS    |        |                      |         | HUBREF   | HL10     | HL4      | VCC1_8            | HL9     | J |
| VSS    |        |                      |         |          | VCC1_8   | VSS      | HL5               | HL6     | K |
|        |        |                      |         | SDA2     | SDCS1#   | VCC1_8   | SDCS3#            | HL7     | L |
|        |        |                      |         | SDDACK#  | VCC3_3   | SDA1     | SDA0              | HLCOMP  | M |
|        |        | PIORDY               | PDCS1#  | VCC3_3   | IRQ15    | SDIOW#   | SDIOR#            | SIORDY  | N |
| PDD15  | IRQ14  | PDA2                 | SDD6    | SDD10    | SDD0     | SDD15    | SDDREQ            | P       |   |
| PDD0   | PDIOR# | PDA0                 | VCC3_3  | SDD5     | SDD13    | SDD1     | SDD14             | R       |   |
| PDD14  | PDIOW# | PDA1                 | SDD7    | SDD9     | SDD11    | VCC3_3   | SDD2              | T       |   |
| VCC3_3 | PDDREQ | PDDACK#              | PDCS3#  | SDD8     | SDD4     | SDD3     | SDD12             | U       |   |

**NOTES:**

1. PERR#,REQ4#, REQ5#, GNT4#, GNT5#, SUSCLK, HL[11], ALERTCLK, and ALERTDATA are on the ICH (82801AA only). These signals are not on the ICH0 (82801AB).

The non-multiplexed signals in this group (REQ4#, GNT4, and HL[11]) are Reserved on ICH0 (No Connect).

Table 15-1. Ballout list by Interface

| Ball Number   | Signal Names |
|---------------|--------------|
| PCI interface |              |
| G2            | AD0          |
| G4            | AD1          |
| F2            | AD2          |
| F3            | AD3          |
| F4            | AD4          |
| F5            | AD5          |
| E1            | AD6          |
| E2            | AD7          |
| D1            | AD8          |
| D3            | AD9          |
| E4            | AD10         |
| C2            | AD11         |
| C1            | AD12         |
| B1            | AD13         |
| D4            | AD14         |
| C3            | AD15         |
| A4            | AD16         |
| B4            | AD17         |
| C5            | AD18         |
| C6            | AD19         |
| B5            | AD20         |
| E7            | AD21         |
| A6            | AD22         |
| B6            | AD23         |
| D7            | AD24         |
| B8            | AD25         |
| A7            | AD26         |
| A8            | AD27         |
| B7            | AD28         |
| C9            | AD29         |
| D8            | AD30         |

Table 15-1. Ballout list by Interface

| Ball Number | Signal Names                                    |
|-------------|---|
| C7          | AD31  |
| D2          | C/BE[0]#  |
| B2          | C/BE[1]#  |
| A3          | C/BE[2]#  |
| D6          | C/BE[3]#  |
| D9          | DEVSEL#   |
| B3          | FRAME#  |
| A2          | IRDY#   |
| C4          | TRDY#   |
| D5          | STOP#   |
| A9          | PAR   |
| A14         | REQ[0]#   |
| B13         | REQ[1]#   |
| B12         | REQ[2]#   |
| D12         | REQ[3]#   |
| B11         | REQ[4]# (ICH only)                              |
| P4          | REQ[5]#/REQ[B]/GPIO[1]<br>(REQ5# on ICH only)   |
| A13         | GNT[0]#   |
| C13         | GNT[1]#   |
| A12         | GNT[2]#   |
| C12         | GNT[3]#   |
| A11         | GNT[4]# (ICH only)                              |
| R5          | GNT[5]#/GNT[B]#/GPIO[17]<br>(REQ5# on ICH only) |
| C14         | PCICLK  |
| J5          | PCIRST#   |
| B9          | PLOCK#  |
| A1          | SERR#   |
| E9          | PERR# (ICH only)                                |
| K1          | PME#  |
| N6          | REQ[A]#/GPIO[0]                                 |
| P4          | REQ[B]#/GPIO[1]                                 |
| P5          | GNT[A]#/GPIO[16]                                |
| R5          | GNT[B]#/GPIO[17]                                |

**Table 15-1. Ballout list by Interface**

| Ball Number          | Signal Names |
|----------------------|--------------|
| <b>IDE interface</b> |              |
| R10                  | PDD[0]       |
| N9                   | PDD[1]       |
| R9                   | PDD[2]       |
| U9                   | PDD[3]       |
| R8                   | PDD[4]       |
| U8                   | PDD[5]       |
| R7                   | PDD[6]       |
| U7                   | PDD[7]       |
| P7                   | PDD[8]       |
| N7                   | PDD[9]       |
| T8                   | PDD[10]      |
| P8                   | PDD[11]      |
| T9                   | PDD[12]      |
| P9                   | PDD[13]      |
| T10                  | PDD[14]      |
| P10                  | PDD[15]      |
| P15                  | SDD[0]       |
| R16                  | SDD[1]       |
| T17                  | SDD[2]       |
| U16                  | SDD[3]       |
| U15                  | SDD[4]       |
| R14                  | SDD[5]       |
| P13                  | SDD[6]       |
| T13                  | SDD[7]       |
| U14                  | SDD[8]       |
| T14                  | SDD[9]       |
| P14                  | SDD[10]      |
| T15                  | SDD[11]      |
| U17                  | SDD[12]      |
| R15                  | SDD[13]      |
| R17                  | SDD[14]      |
| P16                  | SDD[15]      |
| N12                  | PDCS1#       |
| L14                  | SDCS1#       |
| U13                  | PDCS3#       |
| L16                  | SDCS3#       |
| R12                  | PDA[0]       |
| T12                  | PDA[1]       |

**Table 15-1. Ballout list by Interface**

| Ball Number                        | Signal Names                                  |
|------------------------------------|---|
| P12                                | PDA[2]  |
| M16                                | SDA[0]  |
| M15                                | SDA[1]  |
| L13                                | SDA[2]  |
| U11                                | PDDREQ  |
| P17                                | SDDREQ  |
| U12                                | PDDACK#                                       |
| M13                                | SDDACK#                                       |
| R11                                | PDIOR#  |
| N16                                | SDIOR#  |
| T11                                | PDIOW#  |
| N15                                | SDIOW#  |
| N11                                | PIORDY  |
| N17                                | SIORDY  |
| <b>CPU interface</b>               |   |
| F13                                | A20M#   |
| E12                                | CPUSLP#                                       |
| F15                                | FERR#   |
| B17                                | IGNNE#  |
| E15                                | INIT#   |
| E14                                | INTR  |
| B16                                | NMI   |
| F14                                | SMI#  |
| A17                                | STPCLK#                                       |
| A15                                | RCIN#   |
| B15                                | A20GATE                                       |
| <b>SMBUS interface</b>             |   |
| J1                                 | SMBDATA                                       |
| J2                                 | SMBCLK  |
| M1                                 | SMBALERT#/GPIO[11]                            |
| <b>System Management interface</b> |   |
| J4                                 | INTRUDER#/GPIO[10]                            |
| M5                                 | ALERTCLK/GPIO[27]<br>(ALERTCLK on ICH only)   |
| L5                                 | ALERTDATA/GPIO[28]<br>(ALERTDATA on ICH only) |

Table 15-1. Ballout list by Interface

| Ball Number                      | Signal Names       |
|----------------------------------|--------------------|
| <b>Interrupt Interface</b>       |                    |
| R4                               | SERIRQ             |
| D10                              | PIRQ[A]#           |
| A10                              | PIRQ[B]#           |
| B10                              | PIRQ[C]#           |
| C10                              | PIRQ[D]#           |
| P11                              | IRQ[14]            |
| N14                              | IRQ[15]            |
| E16                              | APICD[0]           |
| C17                              | APICD[1]           |
| C16                              | APICCLK            |
| <b>Real Time Clock Interface</b> |                    |
| H3                               | RTCX1              |
| H4                               | RTCX2              |
| <b>AC'97 LINK</b>                |                    |
| T1                               | AC_RST#            |
| T3                               | AC_SYNC            |
| R3                               | AC_BIT_CLK         |
| T2                               | AC_SDOUT           |
| U1                               | AC_SDIN[0]         |
| P3                               | AC_SDIN[1]/GPIO[9] |
| <b>LPC/FWH Interface</b>         |                    |
| R6                               | LAD[0]/FWH[0]      |
| U5                               | LAD[1]/FWH[1]      |
| T5                               | LAD[2]/FWH[2]      |
| T4                               | LAD[3]/FWH[3]      |
| U4                               | LFRAME#/FWH[4]     |
| T6                               | LDRQ[0]#           |
| N3                               | LDRQ[1]#/GPIO[8]   |
| <b>USB Interface</b>             |                    |
| R1                               | USB1P              |
| P2                               | USBP1N             |
| P1                               | USBP0P             |
| N2                               | USBP0N             |
| M3                               | OC[0]#             |
| M4                               | OC[1]#             |

Table 15-1. Ballout list by Interface

| Ball Number                  | Signal Names                                 |
|------------------------------|--|
| <b>Power Management</b>      |  |
| D14                          | THRM#  |
| K3                           | SLP_S3#/GPIO[24]                             |
| K2                           | SLP_S5#                                      |
| J3                           | PWROK  |
| M2                           | PWRBTN#                                      |
| L3                           | RI#  |
| F1                           | RSMRST#                                      |
| L4                           | SUS_STAT#/GPIO[25]                           |
| K4                           | SUSCLK/GPIO[26]<br>(SUSCLK is on ICH only)   |
| <b>Other Clocks</b>          |  |
| U6                           | CLK14  |
| U2                           | CLK48  |
| A16                          | CLK66  |
| <b>Unmuxed GPIO</b>          |  |
| D11                          | GPIO[5]                                      |
| E11                          | GPIO[6]                                      |
| E9                           | GPIO[7] (unmuxed on ICH0)<br>(muxed on ICH)  |
| N4                           | GPIO[12]                                     |
| L2                           | GPIO[13]                                     |
| B14                          | GPIO[21]                                     |
| D13                          | GPIO[22]                                     |
| D15                          | GPIO[23]                                     |
| K4                           | GPIO[26] (unmuxed on ICH0)<br>(muxed on ICH) |
| M5                           | GPIO[27] (unmuxed on ICH0)<br>(muxed on ICH) |
| L5                           | GPIO[28] (unmuxed on ICH0)<br>(muxed on ICH) |
| <b>Miscellaneous Signals</b> |  |
| U3                           | SPKR   |
| H1                           | RTCST#                                       |

**Table 15-1. Ballout list by Interface**

| Ball Number          | Signal Names                             |
|----------------------|--|
| <b>HUB Interface</b> |  |
| D17                  | HL[0]                                    |
| E17                  | HL[1]                                    |
| F17                  | HL[2]                                    |
| G16                  | HL[3]                                    |
| J15                  | HL[4]                                    |
| K16                  | HL[5]                                    |
| K17                  | HL[6]                                    |
| L17                  | HL[7]                                    |
| H15                  | HL[8]                                    |
| J17                  | HL[9]                                    |
| J14                  | HL[10]                                   |
| F16                  | HL[11] (ICH only)                        |
| G17                  | HL_STB                                   |
| H17                  | HL_STB#                                  |
| M17                  | HLCOMP                                   |
| <b>Reserved Pins</b> |  |
| A11                  | Reserved (ICH0 only)<br>(GNT[4]# on ICH) |
| B11                  | Reserved (ICH0 only)<br>(REQ[4]# on ICH) |
| F16                  | Reserved (ICH0 only)<br>(HL[11] on ICH)  |

**Table 15-1. Ballout list by Interface**

| Ball Number   | Signal Names |
|---|--------------|
| <b>Grounds</b>  |              |
| G3, G14, H8,<br>H9, H10, J8,<br>J9, J10, K8,<br>K9, K10, K15,<br>R2                       | GND          |
| <b>Power Pins</b>   |              |
| A5, C8, C11,<br>D16, E13, E6,<br>E5, E3, G5,<br>M14, N5,<br>N13, P6, R13,<br>T7, T16, U10 | VCC3_3       |
| G13, G15,<br>H16, H14,<br>J16, K14, L15,  | VCC1_8       |
| L1, N1  | VCC_SUS      |
| G1  | VCCRTC       |
| H2  | VBIAS        |
| J13   | HUBREF       |
| C15   | 5VREF        |

**NOTES:**

- PERR#, REQ4#, REQ5#, GNT4#, GNT5#, SUSCLK, HL[11], ALERTCLK, and ALERTDATA are on the ICH (82801AA only). These signals are not on the ICH0 (82801AB).

The non-multiplexed signals in this group (REQ4#, GNT4#, and HL[11]) are Reserved on ICH0 (No Connect).

## 15.2 Package Information

Figure 15-3, Figure 15-4, and Table 15-2 show the package dimensions for the ICH and ICH0 (Intel® 82801AA and Intel® 82801AB).

Figure 15-3. Package Dimensions (241 BGA) – Top and Side Views

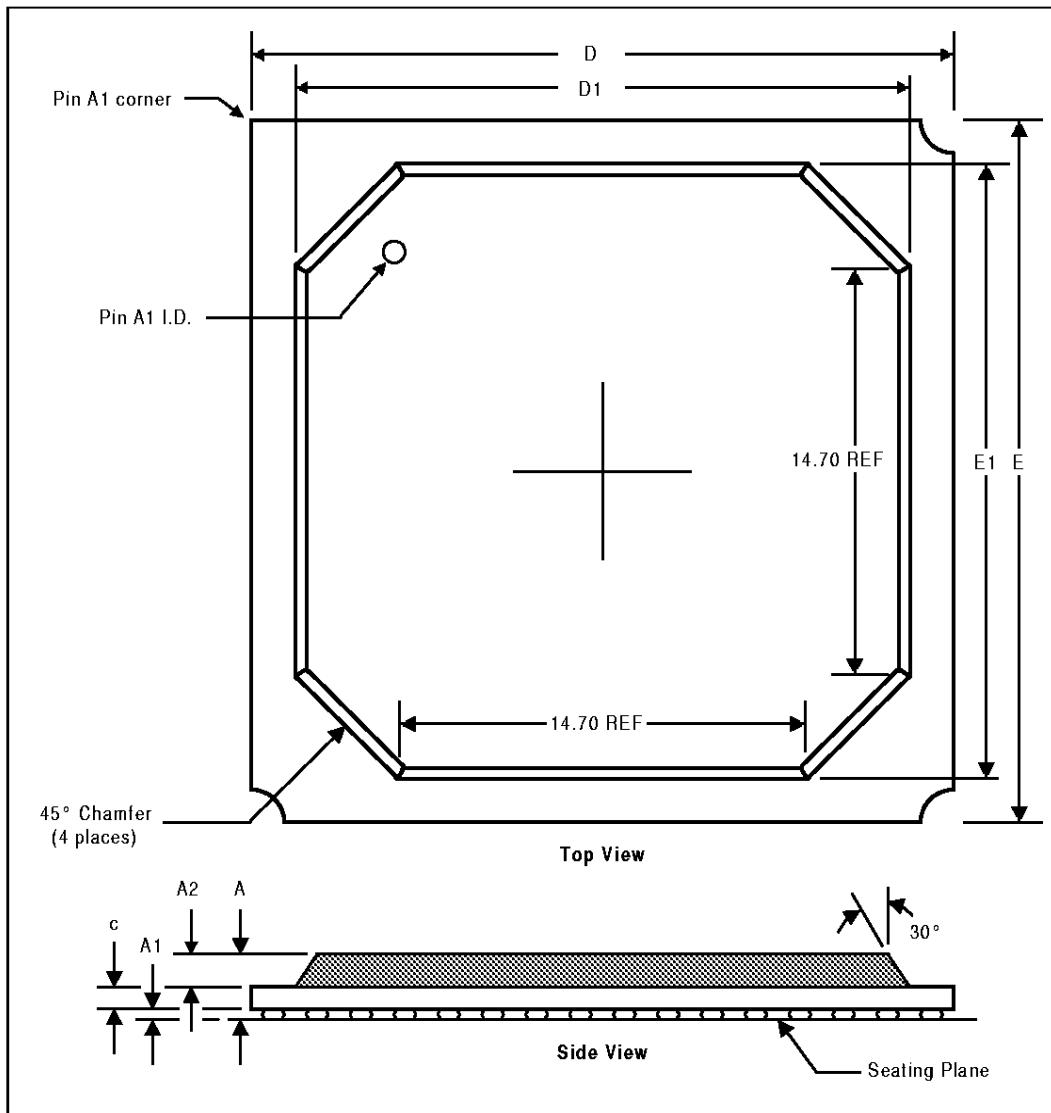


Figure 15-4. ICH Package Dimensions (241 BGA) – Bottom View

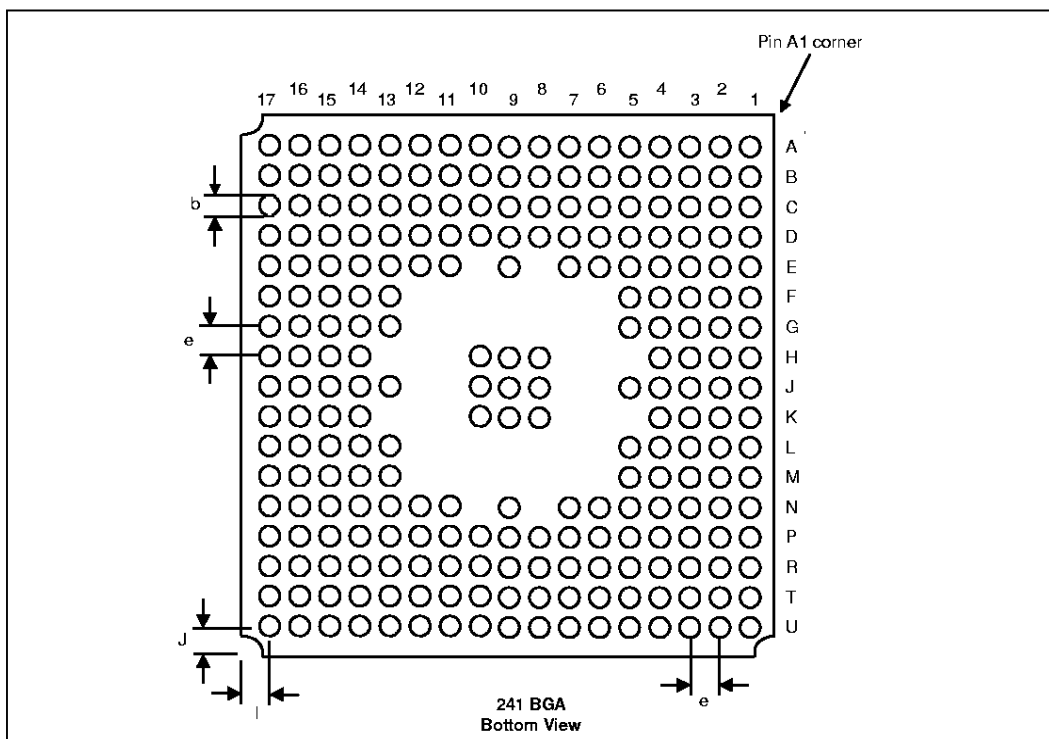


Table 15-2. Package Dimensions (241 BGA)

| Symbol         | Min                      | Nominal | Max   | Units | Note |
|----------------|--------------------------|---------|-------|-------|------|
| A              | 2.14                     | 2.33    | 2.52  | mm    |      |
| A1             | 0.50                     | 0.60    | 0.70  | mm    |      |
| A2             | 1.12                     | 1.17    | 1.22  | mm    |      |
| D              | 22.90                    | 23.00   | 24.00 | mm    |      |
| D1             | 19.30                    | 19.50   | 19.70 | mm    |      |
| E              | 22.90                    | 23.00   | 24.00 | mm    |      |
| E1             | 19.30                    | 19.50   | 19.70 | mm    |      |
| e              | 1.27 (solder ball pitch) |         |       | mm    |      |
| l              | 1.34 REF.                |         |       | mm    |      |
| J              | 1.34 REF.                |         |       | mm    |      |
| M              | 17 x 17 Matrix           |         |       | mm    |      |
| b <sup>2</sup> | 0.60                     | 0.75    | 0.90  | mm    |      |
| c              | 0.52                     | 0.56    | 0.60  | mm    |      |
| e              | 1.27                     |         |       | mm    |      |

**NOTES:**

1. All dimensions and tolerances conform to ANSI Y14.5-1982
2. Dimension is measured at maximum solder ball diameter parallel to primary datum (-C-)
3. Primary Datum (-C-) and seating plane are defined by the spherical crowns of the solder balls.





# Testability

# 16

## 16.1 Test Mode Description

The ICH supports two types of test modes, a tri-state test mode and a NAND tree test mode. Driving RTCRST# low for a specific number of PCI clocks while PWROK is high will activate a particular test mode as described in Table 16-1.

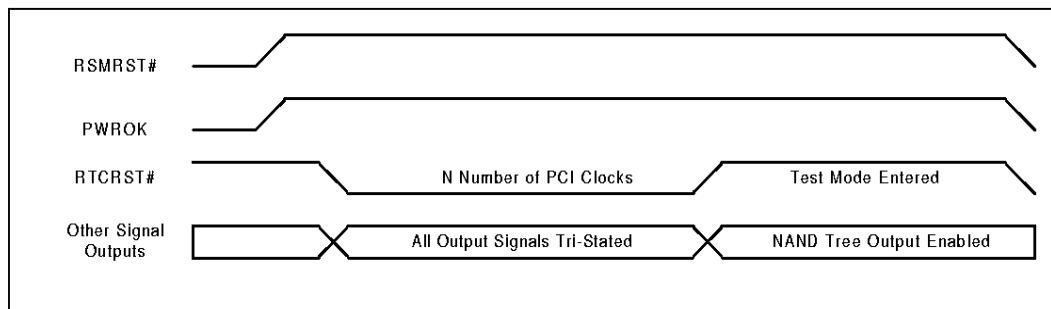
**Note:** RTCRST# can be driven anytime after PCIRST# is inactive; but it is better to drive it as soon as PWROK goes high to enter a test mode

Table 16-1. Test Mode Selection

| Number of PCI Clocks RTCRST# driven low after PWROK active | Test Mode                |
|--|--------------------------|
| <4   | No Test Mode Selected    |
| 4  | NAND Tree 1              |
| 5  | NAND Tree 2              |
| 6  | NAND Tree 3              |
| 7  | NAND Tree 4              |
| 8  | All "Z"                  |
| 9  | No Test Mode Selected    |
| 10 - 24  | Reserved. DO NOT ATTEMPT |
| >24  | No Test Mode Selected    |

Figure 16-1 illustrates the entry into a test mode. A particular test mode is entered upon the rising edge of the RTCRST# after being asserted for a specific number of PCI clocks while PWROK is active. To change test modes, the same sequence should be followed again. To restore the ICH to normal operation, execute the sequence with RTCRST# being asserted so that no test mode is selected as specified in Table 16-1.

Figure 16-1. Test Mode Entry (NAND Tree Example)



## 16.2 Tri-state Mode

When in the tri-state mode, all outputs and bi-directional pin are tri-stated, including the NAND tree outputs.

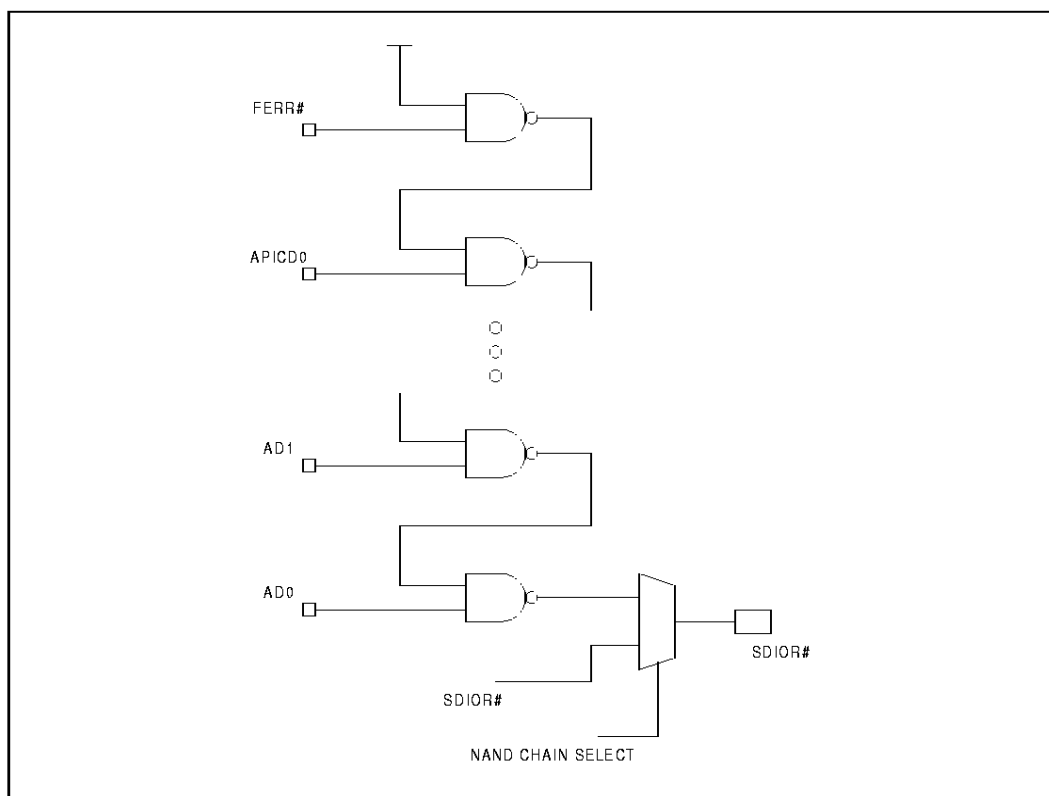
## 16.3 NAND tree Mode

The ICH has four independent NAND trees. Each one can be enabled by itself. When a NAND tree is enabled, all output and bi-directional buffers within that tree are tri-stated, except for the NAND tree output. All output and bi-directional buffers for pins not in the selected NAND tree are tri-stated. Every output signal, except for each NAND tree's output buffer, is configured as an input and is included in the NAND tree.

Table 16-2 - Table 16-5 list each NAND tree pin ordering, with the first value being the first input and the last value being the NAND tree output. Table 16-6 lists the signal pins not included in any NAND tree.

To perform a NAND tree test, drive all NAND tree input pins to 0; the output of the NAND tree will be a 1. Starting at the last signal in the NAND tree (signal at the bottom of list next to output), drive a 1 to each signal, one at a time. The NAND tree output will toggle on each input signal transitioned from 0 to 1. The input-to-output propagation delay specification is 500 ns. Figure 16-2 is a schematic of a NAND tree circuitry.

Figure 16-2. NAND Tree Circuitry (NAND Tree #1 Example)



**Table 16-2. NAND Tree #1 (RTCRST# asserted for 4 PCI clocks while PWROK active)**

| Pin Name                               | Ball # | Notes              | Pin Name | Ball # | Notes                  |
|--|--------|--------------------|----------|--------|------------------------|
| HL_STB                                 | G17    | Top of NAND Tree   | AD16     | A4     |                        |
| REQ[1]#                                | B13    | 2nd signal in NAND | AD18     | C5     |                        |
| GNT[0]#                                | A13    |                    | C/BE[3]# | D6     |                        |
| GNT[3]#                                | C12    |                    | C/BE[2]# | A3     |                        |
| GPIO[6]                                | E11    |                    | AD17     | B4     |                        |
| REQ[2]#                                | B12    |                    | STOP#    | D5     |                        |
| GPIO[5]                                | D11    |                    | TRDY#    | C4     |                        |
| GNT[2]#                                | A12    |                    | SERR#    | A1     |                        |
| PIRQA#                                 | D10    |                    | IRDY#    | A2     |                        |
| <b>ICH (82801AA):</b><br>REQ[4]#       | B11    |                    | AD14     | D4     |                        |
| <b>ICH0 (82801AB):</b><br>Reserved     |        |                    |          |        |                        |
| <b>ICH (82801AA):</b><br>GNT[4]#       | A11    |                    | FRAME#   | B3     |                        |
| <b>ICH0 (82801AB):</b><br>Reserved     |        |                    |          |        |                        |
| PIRQC#                                 | B10    |                    | AD10     | E4     |                        |
| PIRQD#                                 | C10    |                    | AD15     | C3     |                        |
| PIRQB#                                 | A10    |                    | AD5      | F5     |                        |
| PAR                                    | A9     |                    | AD11     | C2     |                        |
| DEVSEL#                                | D9     |                    | C/BE[1]# | B2     |                        |
| <b>ICH (82801AA):</b><br>PERR#/GPIO[7] | E9     |                    | AD13     | B1     |                        |
| <b>ICH0 (82801AB):</b><br>GPIO[7]      |        |                    |          |        |                        |
| PLOCK#                                 | B9     |                    | AD9      | D3     |                        |
| AD29                                   | C9     |                    | C/BE[0]# | D2     |                        |
| AD27                                   | A8     |                    | AD12     | C1     |                        |
| AD25                                   | B8     |                    | AD4      | F4     |                        |
| AD26                                   | A7     |                    | AD3      | F3     |                        |
| AD28                                   | B7     |                    | AD7      | E2     |                        |
| AD30                                   | D8     |                    | AD2      | F2     |                        |
| AD31                                   | C7     |                    | AD8      | D1     |                        |
| AD24                                   | D7     |                    | AD6      | E1     |                        |
| AD23                                   | B6     |                    | AD1      | G4     |                        |
| AD21                                   | E7     |                    | AD0      | G2     | Last in NAND Tree      |
| AD22                                   | A6     |                    | SDIOR#   | N16    | NAND TREE<br>#1 OUTPUT |
| AD20                                   | B5     |                    |          |        |                        |
| AD19                                   | C6     | Connected to AD16  |          |        |                        |

**Table 16-3. NAND TREE #2 (RTCRST# asserted for 5 PCI clocks while PWROK active)**

| Pin Name  | Ball # | Notes                  |
|---|--------|------------------------|
|   |        |                        |
| PCIRST#   | J5     | Top of NAND Tree       |
| SMBCLK  | J2     | 2nd signal in NAND     |
| SMBDATA   | J1     |                        |
| GPIO24/SLP_S3#  | K3     |                        |
| PME#  | K1     |                        |
| RI#   | L3     |                        |
| SLP_S5#   | K2     |                        |
| <b>ICH (82801AA):</b><br>GPIO26/SUSCLK<br><b>ICH0 (82801AB):</b><br>GPIO[26]        | K4     |                        |
| GPIO11/<br>SMBALERT#  | M1     |                        |
| GPIO25/<br>SUS_STAT#  | L4     |                        |
| GPIO13  | L2     |                        |
| <b>ICH (82801AA):</b><br>GPIO28/<br>ALERTDATA<br><b>ICH0 (82801AB):</b><br>GPIO[28] | L5     |                        |
| USBP0P  | P1     | Connected to<br>USBP0N |

| Pin Name   | Ball # | Notes                      |
|--|--------|----------------------------|
| USBP0N   | N2     |                            |
| OC0#   | M3     |                            |
| USBP1P   | R1     |                            |
| USBP1N   | P2     |                            |
| OC1#   | M4     |                            |
| AC_RST#  | T1     |                            |
| GPIO9/AC_SDIN1   | P3     |                            |
| PWRBTN#  | M2     |                            |
| GPIO12   | N4     |                            |
| <b>ICH (82801AA):</b><br>GPIO27/<br>ALERTCLK<br><b>ICH0 (82801AB):</b><br>GPIO[27] | M5     |                            |
| AC_SDIN0   | U1     |                            |
| GPIO8/LDRQ1  | N3     | Last in NAND Tree          |
| PDIOW#   | T11    | NAND TREE<br><br>#2 OUTPUT |

**Table 16-4. NAND Tree #3 (RTCRST# asserted for 6 PCI clocks while PWROK active)**

| Pin Name   | Ball # | Notes              |
|--|--------|--------------------|
| HL_STB#  | H17    | Top of NAND Tree   |
| HL2  | F17    | 2nd signal in NAND |
| HL3  | G16    |                    |
| HL1  | E17    |                    |
| HL0  | D17    |                    |
| <b>ICH (82801 AA):</b><br>HL11<br><b>ICH0 (82801 AB):</b><br>Reserved] | F16    |                    |
| FERR#  | F15    |                    |
| APICD0   | E16    |                    |
| APICD1   | C17    |                    |
| SMI#   | F14    |                    |
| IGNNE#   | B17    |                    |
| INIT#  | E15    |                    |
| STPCLK#  | A17    |                    |
| INTR   | E14    |                    |
| A20M#  | F13    |                    |
| NMI  | B16    |                    |
| GPIO23   | D15    |                    |
| THRM#  | D14    |                    |
| CPUSLP#  | E12    |                    |
| RC_IN#   | A15    |                    |
| A20GATE  | B15    |                    |
| GPIO22   | D13    |                    |
| GPIO21   | B14    | Connected to REQ0# |

| Pin name | Ball # | Notes                  |
|----------|--------|------------------------|
| REQ[0]#  | A14    |                        |
| REQ[3]#  | D12    |                        |
| GNT[1]#  | C13    |                        |
| SDD15    | P16    |                        |
| SDDREQ   | P17    |                        |
| SDIOR#   | N16    |                        |
| SDA1     | M15    |                        |
| SDA0     | M16    |                        |
| SDA2     | L13    |                        |
| SIORDY   | N17    |                        |
| SDCS1#   | L14    |                        |
| SDCS3#   | L16    |                        |
| HLCOMP   | M17    |                        |
| HL5      | K16    |                        |
| HL7      | L17    |                        |
| HL6      | K17    |                        |
| HL4      | J15    |                        |
| HL10     | J14    |                        |
| HL9      | J17    |                        |
| HL8      | H15    | Last in NAND Tree      |
| SDIOW#   | N15    | NAND TREE<br>#3 OUTPUT |

Table 16-5. NAND Tree #4 (RTCRST# asserted for 7 PCI clocks while PWROK active)

| Pin Name  | Ball # | Notes               | Pin Name | Ball # | Notes                  |
|---|--------|---------------------|----------|--------|------------------------|
| AC_SDOUT  | T2     | Top of NAND Tree    | PDIOW#   | T11    |                        |
| AC_BITCLK                                       | R3     | 2nd signal in NAND  | PDD15    | P10    |                        |
| <b>ICH (82801AA):</b><br>GPIO1/REQ[B]#/REQ[5]   | P4     |                     | PDDACK#  | U12    |                        |
| <b>ICH0 (82801AB):</b><br>GPIO1/REQ[B]#         |        |                     |          |        |                        |
| GPIO16/GNT[A]#                                  | P5     |                     | IRQ14    | P11    |                        |
| SERIRQ  | R4     |                     | PDA1     | T12    |                        |
| GPIO0/REQ[A]#                                   | N6     |                     | PIORDY   | N11    |                        |
| SPKR  | U3     |                     | PDA0     | R12    |                        |
| AC_SYNC   | T3     |                     | PDCS3#   | U13    |                        |
| LAD[3]/FWH[3]                                   | T4     |                     | SDD7     | T13    |                        |
| <b>ICH (82801AA):</b><br>GPIO17/GNT[B]#/GNT[5]# | R5     |                     | SDD8     | U14    |                        |
| <b>ICH0 (82801AB):</b><br>GPIO17/GNT[B]#        |        |                     |          |        |                        |
| LFRAME#/FWH[4]                                  | U4     |                     | SDD9     | T14    |                        |
| LAD[2]/FWH[2]                                   | T5     |                     | PDA2     | P12    |                        |
| LAD[0]/FWH[0]                                   | R6     |                     | SDD5     | R14    |                        |
| LAD[1]/FWH[1]                                   | U5     |                     | SDD11    | T15    |                        |
| LDRQ[0]#  | T6     |                     | SDD4     | U15    |                        |
| PDD9  | N7     |                     | PDCS1#   | N12    |                        |
| PDD8  | P7     |                     | SDD13    | R15    |                        |
| PDD6  | R7     |                     | SDD3     | U16    |                        |
| PDD11   | P8     |                     | SDD6     | P13    |                        |
| PDD7  | U7     |                     | SDD1     | R16    |                        |
| PDD5  | U8     |                     | SDD12    | U17    |                        |
| PDD4  | R8     |                     | SDDACK#  | M13    |                        |
| PDD10   | T8     |                     | SDD2     | T17    |                        |
| PDD3  | U9     |                     | SDD0     | P15    |                        |
| PDD13   | P9     |                     | SDD10    | P14    |                        |
| PDD1  | N9     |                     | IRQ15    | N14    |                        |
| PDD12   | T9     |                     | SDD14    | R17    |                        |
| PDD2  | R9     |                     | SDIOW#   | N15    | Last in NAND Tree      |
| PDD14   | T10    |                     | PDIOR#   | R11    | NAND TREE<br>#4 OUTPUT |
| PDD0  | R10    |                     |          |        |                        |
| PDDREQ  | U11    | Connected to PDIOW# |          |        |                        |

**Table 16-6. Signals Not in NAND Tree**

| Pin Name | Ball # | Notes |
|----------|--------|-------|
| RSMRST#  | F1     |       |
| PWROK    | J3     |       |
| RTCX1    | H3     |       |
| RTCX2    | H4     |       |
| VBIAS    | H2     |       |
| RTCRST#  | H1     |       |

| Pin Name | Ball # | Notes |
|----------|--------|-------|
| CLK14    | U6     |       |
| CLK48    | U2     |       |
| CLK66    | A16    |       |
| APICCLK  | C16    |       |
| PCICLK   | C14    |       |
| PDIOR#   | R11    |       |





# Register Index

# A

Table A-1. ICH PCI Configuration Registers (Sheet 1 of 6)

| Register Name                  | Offset  | Datasheet Section and Location  |
|--------------------------------|---------|---|
| Hub Link to PCI Bridge D30:F0  |         |   |
| Vendor ID                      | 00h–01h | Section 7.1.1, "VID—Vendor ID Register (HUB-PCI—D30:F0)" on page 7-2                            |
| Device ID                      | 02h–03h | Section 7.1.2, "DID—Device ID Register (HUB-PCI—D30:F0)" on page 7-2                            |
| PCI Device Command Register    | 04h–05h | Section 7.1.3, "CMD—Command Register (HUB-PCI—D30:F0)" on page 7-3                              |
| PCI Device Status Register     | 06h–07h | Section 7.1.4, "PD_STS—Primary Device Status Register (HUB-PCI—D30:F0)" on page 7-4             |
| Revision ID                    | 08h     | Section 7.1.5, "RID—Revision ID Register (HUB-PCI—D30:F0)" on page 7-5                          |
| Sub Class Code                 | 0Ah     | Section 7.1.6, "SCC—Sub-Class Code Register (HUB-PCI—D30:F0)" on page 7-5                       |
| Base Class Code                | 0Bh     | Section 7.1.7, "BCC—Base-Class Code Register (HUB-PCI—D30:F0)" on page 7-5                      |
| Primary Master Latency Timer   | 0Dh     | Section 7.1.8, "PMLT—Primary Master Latency Timer Register (HUB-PCI—D30:F0)" on page 7-5        |
| Header Type                    | 0Eh     | Section 7.1.9, "HEADTYP—Header Type Register (HUB-PCI—D30:F0)" on page 7-6                      |
| Primary Bus Number             | 18h     | Section 7.1.10, "PBUS_NUM—Primary Bus Number Register (HUB-PCI—D30:F0)" on page 7-6             |
| Secondary Bus Number           | 19h     | Section 7.1.11, "SBUS_NUM—Secondary Bus Number Register (HUB-PCI—D30:F0)" on page 7-6           |
| Subordinate Bus Number         | 1Ah     | Section 7.1.12, "SUB_BUS_NUM—Subordinate Bus Number Register (HUB-PCI—D30:F0)" on page 7-6      |
| Secondary Master Latency Timer | 1Bh     | Section 7.1.13, "SMLT—Secondary Master Latency Timer Register (HUB-PCI—D30:F0)" on page 7-7     |
| IO Base Register               | 1Ch     | Section 7.1.14, "IOBASE—I/O Base Register (HUB-PCI—D30:F0)" on page 7-7                         |
| IO Limit Register              | 1Dh     | Section 7.1.15, "IOLIM—I/O Limit Register (HUB-PCI—D30:F0)" on page 7-7                         |
| Secondary Status Register      | 1Eh–1Fh | Section 7.1.16, "SECSTS—Secondary Status Register (HUB-PCI—D30:F0)" on page 7-8                 |
| Memory Base                    | 20h–21h | Section 7.1.17, "MEMBASE—Memory Base Register (HUB-PCI—D30:F0)" on page 7-9                     |
| Memory Limit                   | 22h–23h | Section 7.1.18, "MEMLIM—Memory Limit Register (HUB-PCI—D30:F0)" on page 7-9                     |
| Prefetchable Memory Base       | 24h–25h | Section 7.1.19, "PREF_MEM_BASE—Prefetchable Memory Base Register (HUB-PCI—D30:F0)" on page 7-9  |
| Prefetchable Memory Limit      | 26h–27h | Section 7.1.20, "PREF_MEM_MLT—Prefetchable Memory Limit Register (HUB-PCI—D30:F0)" on page 7-10 |
| I/O Base Upper 16 Bits         | 30h–31h | Section 7.1.21, "IOBASE_HI—I/O Base Upper 16 Bits Register (HUB-PCI—D30:F0)" on page 7-10       |

Table A-1. ICH PCI Configuration Registers (Sheet 2 of 6)

| Register Name                   | Offset  | Datasheet Section and Location  |
|---------------------------------|---------|---|
| I/O Limit Upper 16 Bits         | 32h–33h | Section 7.1.22, "IOLIM_HI—I/O Limit Upper 16 Bits Register (HUB-PCI—D30:F0)" on page 7-10   |
| Interrupt Line                  | 3Ch     | Section 7.1.23, "INT_LINE—Interrupt Line Register (HUB-PCI—D30:F0)" on page 7-10            |
| Bridge Control                  | 3Eh–3Fh | Section 7.1.24, "BRIDGE_CNT—Bridge Control Register (HUB-PCI—D30:F0)" on page 7-11          |
| ICH Configuration Register      | 50h–51h | Section 7.1.25, "CNF—ICH Configuration Register (HUB-PCI—D30:F0)" on page 7-12              |
| Multi-Transaction Timer         | 70h     | Section 7.1.26, "MTT—Multi-Transaction Timer Register (HUB-PCI—D30:F0)" on page 7-13        |
| PCI Master Status               | 82h     | Section 7.1.27, "PCI_MAST_STS—PCI Master Status Register (HUB-PCI—D30:F0)" on page 7-13     |
| Error Command Register          | 90h     | Section 7.1.28, "ERR_CMD—Error Command Register (HUB-PCI—D30:F0)" on page 7-14              |
| Error Status Register           | 92h     | Section 7.1.29, "ERR_STS—Error Status Register (HUB-PCI—D30:F0)" on page 7-14               |
| <b>LPC Bridge D31:F0</b>        |         |   |
| Vendor ID                       | 00h–01h | Section 8.1.1, "VID—Vendor ID Register (LPC I/F—D31:F0)" on page 8-2                        |
| Device ID                       | 02h–03h | Section 8.1.2, "DID—Device ID Register (LPC I/F—D31:F0)" on page 8-2                        |
| PCI Command Register            | 04h–05h | Section 8.1.3, "PCICMD—PCI COMMAND Register (LPC I/F—D31:F0)" on page 8-3                   |
| PCI Device Status Register      | 06h–07h | Section 8.1.4, "PCISTA—PCI Device Status (LPC I/F—D31:F0)" on page 8-4                      |
| Revision ID                     | 08h     | Section 8.1.5, "RID—Revision ID Register (LPC I/F—D31:F0)" on page 8-5                      |
| Programming Interface           | 09h     | Section 8.1.6, "PI—Programming Interface (LPC I/F—D31:F0)" on page 8-5                      |
| Sub Class Code                  | 0Ah     | Section 8.1.7, "SCC—Sub-Class Code Register (LPC I/F—D31:F0)" on page 8-5                   |
| Base Class Code                 | 0Bh     | Section 8.1.8, "BCC—Base-Class Code Register (LPC I/F—D31:F0)" on page 8-5                  |
| Header Type                     | 0Eh     | Section 8.1.9, "HEADTYP—Header Type Register (LPC I/F—D31:F0)" on page 8-5                  |
| ACPI Base Address Register      | 40h–43h | Section 8.1.10, "PMBASE—ACPI Base Address (LPC I/F—D31:F0)" on page 8-6                     |
| ACPI Control                    | 44h     | Section 8.1.11, "ACPI_CNTL—ACPI Control (LPC I/F—D31:F0)" on page 8-6                       |
| BIOS Control Register           | 4Eh–4Fh | Section 8.1.12, "BIOS_CNTL (LPC I/F—D31:F0)" on page 8-7                                    |
| TCO Control                     | 54h     | Section 8.1.13, "TCO_CNTL—TCO Control (LPC I/F—D31:F0)" on page 8-7                         |
| GPIO Base Address Register      | 58h–5Bh | Section 8.1.14, "GPIOBASE—GPIO Base Address (LPC I/F—D31:F0)" on page 8-8                   |
| GPIO Control Register           | 5Ch     | Section 8.1.15, "GPIO_CNTL—GPIO Control (LPC I/F—D31:F0)" on page 8-8                       |
| PIRQ[A-D] Routing Control       | 60h–63h | Section 8.1.16, "PIRQ[n]_ROUT—PIRQ[A,B,C,D] Routing Control (LPC I/F—D31:F0)" on page 8-9   |
| Serial IRQ Control Register     | 64h     | Section 8.1.17, "SERIRQ_CNTL—Serial IRQ Control (LPC I/F—D31:F0)" on page 8-9               |
| Device 31 Error Config Register | 88h     | Section 8.1.18, "D31_ERR_CFG—Device 31 Error Config Register (LPC I/F—D31:F0)" on page 8-10 |

Table A-1. ICH PCI Configuration Registers (Sheet 3 of 6)

| Register Name                            | Offset  | Datasheet Section and Location   |
|--|---------|--|
| Device 31 Error Status Register          | 8Ah     | Section 8.1.19, "D31_ERR_STS—Device 31 Error Status Register (LPC I/F—D31:F0)" on page 8-10      |
| PCI DMA Configuration Registers          | 90h–91h | Section 8.1.20, "PCI_DMA_CFG—PCI DMA Configuration (LPC I/F—D31:F0)" on page 8-11                |
| General Power Management Configuration 1 | A0h     | Section 8.8.1.1, "GEN_PMCON_1—General PM Configuration 1 Register (PM—D31:F0)" on page 8-52      |
| General Power Management Configuration 2 | A2h     | Section 8.8.1.2, "GEN_PMCON_2—General PM Configuration 2 Register (PM—D31:F0)" on page 8-53      |
| General Power Management Configuration 3 | A4h     | Section 8.8.1.3, "GEN_PMCON_3—General PM Configuration 3 Register (PM—D31:F0)" on page 8-53      |
| GPI_ROUT                                 | B8h–BBh | Section 8.8.1.4, "GPI_ROUT—GPI Routing Control Register (PM—D31:F0)" on page 8-54                |
| IO_MON_RNG1                              | C4h     | Section 8.8.1.5, "IO_MON_RNG1—IO Monitor Range 1 Register (PM—D31:F0)" on page 8-54              |
| IO_MON_RNG2                              | C6h     | Section 8.8.1.6, "IO_MON_RNG2—IO Monitor Range 2 Register (PM—D31:F0)" on page 8-55              |
| IO_MON_MSK                               | CCh     | Section 8.8.1.7, "IO_MON_MSK—IO Monitor Range Mask Register (PM—D31:F0)" on page 8-55            |
| General Control                          | D0h–D3h | Section 8.1.21, "GEN_CNTL — General Control Register (LPC I/F — D31:F0)" on page 8-11            |
| General Status                           | D4h–D7h | Section 8.1.22, "GEN_STA—General Status (LPC I/F—D31:F0)" on page 8-13                           |
| Real Time Clock Configuration            | D8h     | Section 8.1.23, "RTC_CONF—RTC Configuration Register (LPC I/F—D31:F0)" on page 8-14              |
| LPC COM Port Decode Ranges               | E0h     | Section 8.1.24, "COM_DEC—LPC I/F Communication Port Decode Ranges (LPC I/F—D31:F0)" on page 8-14 |
| LPC FDD & LPT Decode Ranges              | E1h     | Section 8.1.25, "FDD/LPT_DEC—LPC I/F FDD & LPT Decode Ranges (LPC I/F—D31:F0)" on page 8-15      |
| LPC Sound Decode Ranges                  | E2h     | Section 8.1.26, "SND_DEC—LPC I/F Sound Decode Ranges (LPC I/F—D31:F0)" on page 8-15              |
| LPC General 1 Decode Range               | E4h–E5h | Section 8.1.27, "GEN1_DEC—LPC I/F Generic Decode Range 1 (LPC I/F—D31:F0)" on page 8-16          |
| LPC General 2 Decode Range               | E6h–E7h | Section 8.1.28, "GEN2_DEC—LPC I/F Generic Decode Range 2 (LPC I/F—D31:F0)" on page 8-16          |
| LPC Enables                              | ECh–EDh | Section 8.1.29, "LPC_EN—LPC I/F Enables (LPC I/F—D31:F0)" on page 8-17                           |
| Function Disable Register                | F2h     | Section 8.1.32, "FUNC_DIS—Function Disable Register (LPC I/F—D31:F0)" on page 8-20               |

Table A-1. ICH PCI Configuration Registers (Sheet 4 of 6)

| Register Name                    | Offset  | Datasheet Section and Location  |
|----------------------------------|---------|---|
| <b>IDE Controller (D31:F1)</b>   |         |   |
| Vendor ID                        | 00h–01h | Section 9.1.1, "VID—Vendor ID Register (IDE—D31:F1)" on page 9-2                          |
| Device ID                        | 02h–03h | Section 9.1.2, "DID—Device ID Register (IDE—D31:F1)" on page 9-2                          |
| Command Register                 | 04h–05h | Section 9.1.3, "CMD—Command Register (IDE—D31:F1)" on page 9-2                            |
| Device Status                    | 06h–07h | Section 9.1.4, "STA—Device Status Register (IDE—D31:F1)" on page 9-3                      |
| Revision ID                      | 08h     | Section 9.1.5, "RID—Revision ID Register (IDE—D31:F1)" on page 9-3                        |
| Programming Interface            | 09h     | Section 9.1.6, "PI—Programming Interface (IDE—D31:F1)" on page 9-3                        |
| Sub Class Code                   | 0Ah     | Section 9.1.7, "SCC—Sub Class Code (IDE—D31:F1)" on page 9-4                              |
| Base Class Code                  | 0Bh     | Section 9.1.8, "BCC—Base Class Code (IDE—D31:F1)" on page 9-4                             |
| Header Type                      | 0Eh     | Section 9.1.10, "HEADTYP—Header Type Register (LPC I/F—D31:F0)" on page 9-4               |
| Base Address Register            | 20h–23h | Section 9.1.11, "BM_BASE—Bus Master Base Address Register (IDE—D31:F1)" on page 9-5       |
| Primary IDE Timing               | 40h–43h | Section 9.1.12, "IDE_TIM—IDE Timing Register (IDE—D31:F1)" on page 9-5                    |
| Slave IDE Timing                 | 44h     | Section 9.1.13, "SLV_IDETIM—Slave (Drive 1) IDE Timing Register (IDE—D31:F1)" on page 9-7 |
| Synchronous DMA Control Register | 48h     | Section 9.1.14, "SDMA_CNT—Synchronous DMA Control Register (IDE—D31:F1)" on page 9-8      |
| Synchronous DMA Timing Register  | 4Ah–4Bh | Section 9.1.15, "SDMA_TIM—Synchronous DMA Timing Register (IDE—D31:F1)" on page 9-9       |
| IDE I/O Configuration Register   | 54h     | Section 9.1.16, "IDE_CONFIG—IDE I/O Configuration Register" on page 9-10                  |
| <b>USB Controller (D31:F2)</b>   |         |   |
| Vendor ID                        | 00h–01h | Section 10.1.1, "VID—Vendor Identification Register (USB—D31:F2)" on page 10-2            |
| Device ID                        | 02h–03h | Section 10.1.2, "DID—Device Identification Register (USB—D31:F2)" on page 10-2            |
| Command Register                 | 04h–05h | Section 10.1.3, "CMD—Command Register (USB—D31:F2)" on page 10-2                          |
| Device Status                    | 06h–07h | Section 10.1.4, "STA—Device Status Register (USB—D31:F2)" on page 10-3                    |
| Revision ID                      | 08h     | Section 10.1.5, "RID—Revision Identification Register (USB—D31:F2)" on page 10-3          |
| Programming Interface            | 09h     | Section 10.1.6, "PI—Programming Interface (USB—D31:F2)" on page 10-3                      |
| Sub Class Code                   | 0Ah     | Section 10.1.7, "SCC—Sub Class Code Register (USB—D31:F2)" on page 10-4                   |
| Base Class Code                  | 0Bh     | Section 10.1.8, "BCC—Base Class Code Register (USB—D31:F2)" on page 10-4                  |
| Header Type                      | 0Eh     | Section 10.1.9, "HEADTYP—Header Type Register (USB—D31:F2)" on page 10-4                  |
| Base Address Register            | 20h–23h | Section 10.1.10, "BASE—Base Address Register (USB—D31:F2)" on page 10-4                   |

Table A-1. ICH PCI Configuration Registers (Sheet 5 of 6)

| Register Name                          | Offset  | Datasheet Section and Location   |
|--|---------|--|
| Interrupt Line                         | 3Ch     | Section 10.1.11, "INTR_LN---Interrupt Line Register (USB---D31:F2)" on page 10-5                       |
| Interrupt Pin                          | 3Dh     | Section 10.1.12, "INTR_PN---Interrupt Pin Register (USB---D31:F2)" on page 10-5                        |
| Serial Bus Release Number              | 60h     | Section 10.1.13, "SB_RELNUM---Serial Bus Release Number Register (USB---D31:F2)" on page 10-5          |
| USB Legacy Keyboard/Mouse Control      | C0h-C1h | Section 10.1.14, "USB_LEGKEY---USB Legacy Keyboard/Mouse Control Register (USB---D31:F2)" on page 10-6 |
| USB Resume Enable                      | C4h     | Section 10.1.15, "USB_RES---USB Resume Enable Register (USB---D31:F2)" on page 10-7                    |
| <b>SMBus Controller (D31:F3)</b>       |         |  |
| Vendor ID                              | 00h-01h | Section 11.1.1, "VID---Vendor Identification Register (SMBUS---D31:F3)" on page 11-2                   |
| Device ID                              | 02h-03h | Section 11.1.2, "DID---Device Identification Register (SMBUS---D31:F3)" on page 11-2                   |
| Command Register                       | 04h-05h | Section 11.1.3, "CMD---Command Register (SMBUS---D31:F3)" on page 11-2                                 |
| Device Status                          | 06h-07h | Section 11.1.4, "STA---Device Status Register (SMBUS---D31:F3)" on page 11-3                           |
| Revision ID                            | 08h     | Section 11.1.5, "RID---Revision ID Register (SMBUS---D31:F3)" on page 11-3                             |
| Programming Interface                  | 09h     | Section 11.1.6, "PI---Programming Interface (SMBUS---D31:F3)" on page 11-3                             |
| Sub Class Code                         | 0Ah     | Section 11.1.7, "SCC---Sub Class Code Register (SMBUS---D31:F3)" on page 11-4                          |
| Base Class Code                        | 0Bh     | Section 11.1.8, "BCC---Base Class Code Register (SMBUS---D31:F3)" on page 11-4                         |
| Header Type                            | 0Eh     | Section 11.1.9, "HEADTYP---Header Type Register (SMBUS---D31:F3)" on page 11-4                         |
| SMB Base Address Register              | 20h-23h | Section 11.1.10, "SMB_BASE---SMBus Base Address Register (SMBUS---D31:F3)" on page 11-4                |
| Interrupt Line                         | 3Ch     | Section 11.1.11, "INTR_LN---Interrupt Line Register (SMBUS---D31:F3)" on page 11-5                     |
| Interrupt Pin                          | 3Dh     | Section 11.1.12, "INTR_PN---Interrupt Pin Register (SMBUS---D31:F3)" on page 11-5                      |
| Host Configuration                     | 40h     | Section 11.1.13, "HOSTC---Host Configuration Register (SMBUS---D31:F3)" on page 11-5                   |
| <b>AC'97 Audio Controller (D31:F5)</b> |         |  |
| Vendor Identification                  | 00h-01h | Section 12.1.1, "VID---Vendor Identification Register (Audio---D31:F5)" on page 12-2                   |
| Device Identification                  | 02h-03h | Section 12.1.2, "DID---Device Identification Register (Audio---D31:F5)" on page 12-2                   |
| PCI Command                            | 04h-05h | Section 12.1.3, "PCICMD---PCI Command Register (Audio---D31:F5)" on page 12-2                          |
| PCI Device Status                      | 06h-07h | Section 12.1.4, "PCISTA---PCI Device Status Register (Audio---D31:F5)" on page 12-3                    |
| Revision Identification                | 08h     | Section 12.1.5, "RID---Revision Identification Register (Audio---D31:F5)" on page 12-3                 |
| Programming Interface                  | 09h     | Section 12.1.6, "PI---Programming Interface Register (Audio---D31:F5)" on page 12-3                    |

Table A-1. ICH PCI Configuration Registers (Sheet 6 of 6)

| Register Name                           | Offset  | Datasheet Section and Location  |
|---|---------|---|
| Sub Class Code                          | 0Ah     | Section 12.1.7, "SCC---Sub Class Code Register (Audio---D31:F5)" on page 12-4                               |
| Base Class Code                         | 0Bh     | Section 12.1.8, "BCC---Base Class Code Register (Audio---D31:F5)" on page 12-4                              |
| Header Type                             | 0Eh     | Section 12.1.9, "HEADTYP---Header Type Register (Audio---D31:F5)" on page 12-4                              |
| Native Audio Mixer Base Address         | 10h–13h | Section 12.1.10, "NAMBAR---Native Audio Mixer Base Address Register (Audio---D31:F5)" on page 12-5          |
| Native Audio Bus Mastering Base Address | 14h–17h | Section 12.1.11, "NABMBAR---Native Audio Bus Mastering Base Address Register (Audio---D31:F5)" on page 12-5 |
| Subsystem Vendor ID                     | 2Ch–2Dh | Section 12.1.12, "SVID---Subsystem Vendor ID Register (Audio---D31:F5)" on page 12-6                        |
| Subsystem ID                            | 2Eh–2Fh | Section 12.1.13, "SID---Subsystem ID Register (Audio---D31:F5)" on page 12-6                                |
| Interrupt Line                          | 3Ch     | Section 12.1.14, "INTR_LN---Interrupt Line Register (Audio---D31:F5)" on page 12-6                          |
| Interrupt Pin                           | 3Dh     | Section 12.1.15, "INTR_PN---Interrupt Pin Register (Audio---D31:F5)" on page 12-7                           |
| <b>AC'97 Modem Controller (D31:F6)</b>  |         |   |
| Vendor Identification                   | 00h–01h | Section 13.1.1, "VID---Vendor Identification Register (Modem---D31:F6)" on page 13-2                        |
| Device Identification                   | 02h–03h | Section 13.1.2, "DID---Device Identification Register (Modem---D31:F6)" on page 13-2                        |
| PCI Command                             | 04h–05h | Section 13.1.3, "PCICMD---PCI Command Register (Modem---D31:F6)" on page 13-2                               |
| PCI Device Status                       | 06h–07h | Section 13.1.4, "PCISTA---Device Status Register (Modem---D31:F6)" on page 13-3                             |
| Revision Identification                 | 08h     | Section 13.1.5, "RID---Revision Identification Register (Modem---D31:F6)" on page 13-3                      |
| Programming Interface                   | 09h     | Section 13.1.6, "PI---Programming Interface Register (Modem---D31:F6)" on page 13-3                         |
| Sub Class Code                          | 0Ah     | Section 13.1.7, "SCC---Sub Class Code Register (Modem---D31:F6)" on page 13-4                               |
| Base Class Code                         | 0Bh     | Section 13.1.8, "BCC---Base Class Code Register (Modem---D31:F6)" on page 13-4                              |
| Header Type                             | 0Eh     | Section 13.1.9, "HEDTYP---Header Type Register (Modem---D31:F6)" on page 13-4                               |
| Modem Mixer Base Address                | 10h–13h | Section 13.1.10, "MMBAR---Modem Mixer Base Address Register (Modem---D31:F6)" on page 13-5                  |
| Modem Base Address                      | 14h–17h | Section 13.1.11, "MBAR---Modem Base Address Register (Modem---D31:F6)" on page 13-5                         |
| Subsystem Vendor ID                     | 2Ch–2Dh | Section 13.1.12, "SVID---Subsystem Vendor ID (Modem---D31:F6)" on page 13-6                                 |
| Subsystem ID                            | 2Eh–2Fh | Section 13.1.13, "SID---Subsystem ID (Modem---D31:F6)" on page 13-6   |
| Interrupt Line                          | 3C      | Section 13.1.14, "INTR_LN---Interrupt Line Register (Modem---D31:F6)" on page 13-6                          |
| Interrupt Pin                           | 3Dh     | Section 13.1.15, "INT_PIN---interrupt Pin (Modem---D31:F6)" on page 13-7                                    |

Table A-2. ICH Fixed I/O Registers (Sheet 1 of 4)

| Register Name  | Port    | Datasheet Section and Location   |
|--|---------|--|
| Channel 0 DMA Base & Current Address Register  | 00h     | Section 8.2.1, "DMABASE_CA—DMA Base and Current Address Registers" on page 8-22  |
| Channel 0 DMA Base & Current Count Register  | 01h     | Section 8.2.2, "DMABASE_CC—DMA Base and Current Count Registers" on page 8-23  |
| Channel 1 DMA Base & Current Address Register  | 02h     | Section 8.2.1, "DMABASE_CA—DMA Base and Current Address Registers" on page 8-22  |
| Channel 1 DMA Base & Current Count Register  | 03h     | Section 8.2.2, "DMABASE_CC—DMA Base and Current Count Registers" on page 8-23  |
| Channel 2 DMA Base & Current Address Register  | 04h     | Section 8.2.1, "DMABASE_CA—DMA Base and Current Address Registers" on page 8-22  |
| Channel 2 DMA Base & Current Count Register  | 05h     | Section 8.2.2, "DMABASE_CC—DMA Base and Current Count Registers" on page 8-23  |
| Channel 3 DMA Base & Current Address Register  | 06h     | Section 8.2.1, "DMABASE_CA—DMA Base and Current Address Registers" on page 8-22  |
| Channel 3 DMA Base & Current Count Register  | 07h     | Section 8.2.2, "DMABASE_CC—DMA Base and Current Count Registers" on page 8-23  |
| Channel 0-3 DMA Command Register<br>Channel 0-3 DMA Status Register  | 08h     | Section 8.2.4, "DMACMD—DMA Command Register" on page 8-24<br>Section 8.2.5, "DMASTA—DMA Status Register" on page 8-24  |
| Channel 0-3 DMA Write Single Mask Register   | 0Ah     | Section 8.2.6, "DMA_WRSMSK—DMA Write Single Mask Register" on page 8-25  |
| Channel 0-3 DMA Channel Mode Register  | 0Bh     | Section 8.2.7, "DMACH_MODE—DMA Channel Mode Register" on page 8-25   |
| Channel 0-3 DMA Clear Byte Pointer Register  | 0Ch     | Section 8.2.8, "DMA Clear Byte Pointer Register" on page 8-26  |
| Channel 0-3 DMA Master Clear Register  | 0Dh     | Section 8.2.9, "DMA Master Clear Register" on page 8-26  |
| Channel 0-3 DMA Clear Mask Register  | 0Eh     | Section 8.2.10, "DMA_CLMSK—DMA Clear Mask Register" on page 8-26   |
| Channel 0-3 DMA Write All Mask Register  | 0Fh     | Section 8.2.11, "DMA_WRMSK—DMA Write All Mask Register" on page 8-27   |
| <b>Aliased at 00h–0Fh</b>  | 10h–1Fh |  |
| Master PIC ICW1 Init. Cmd Word 1 Register<br>Master PIC OCW2 Op Ctrl Word 2 Register<br>Master PIC OCW3 Op Ctrl Word 3 Register  | 20h     | Section 8.4.2, "ICW1—Initialization Command Word 1 Register" on page 8-33<br>Section 8.4.8, "OCW2—Operational Control Word 2 Register" on page 8-36<br>Section 8.4.9, "OCW3—Operational Control Word 3 Register" on page 8-37  |
| Master PIC ICW2 Init. Cmd Word 2 Register<br>Master PIC ICW3 Init. Cmd Word 3 Register<br>Master PIC ICW4 Init. Cmd Word 4 Register<br>Master PIC OCW1 Op Ctrl Word 1 Register | 21h     | Section 8.4.3, "ICW2—Initialization Command Word 2 Register" on page 8-34<br>Section 8.4.4, "ICW3—Master Controller Initialization Command Word 3 Register" on page 8-34<br>Section 8.4.6, "ICW4—Initialization Command Word 4 Register" on page 8-35<br>Section 8.4.7, "OCW1—Operational Control Word 1 (Interrupt Mask) Register" on page 8-35 |
| <b>Aliased at 20h–21h</b>  | 24h–25h |  |
| <b>Aliased at 20h–21h</b>  | 28h–29h |  |
| <b>Aliased at 20h–21h</b>  | 24h–25h |  |
| <b>Aliased at 20h–21h</b>  | 2Ch–2Dh |  |

Table A-2. ICH Fixed I/O Registers (Sheet 2 of 4)

| Register Name   | Port    | Datasheet Section and Location  |
|---|---------|---|
| <b>Aliased at 20h–21h</b>   | 30h–31h |   |
| <b>Aliased at 20h–21h</b>   | 34h–35h |   |
| <b>Aliased at 20h–21h</b>   | 38h–39h |   |
| <b>Aliased at 20h–21h</b>   | 3Ch–3Dh |   |
| Counter 0 Interval Time Status Byte Format<br>Counter 0 Counter Access Port Register          | 40h     | Section 8.3.2, "SBYTE_FMT---Interval Timer Status Byte Format Register" on page 8-30<br>Section 8.3.3, "Counter Access Ports Register" on page 8-31   |
| Counter 1 Interval Time Status Byte Format<br>Counter 1 Counter Access Port Register          | 41h     | Section 8.3.2, "SBYTE_FMT---Interval Timer Status Byte Format Register" on page 8-30<br>Section 8.3.3, "Counter Access Ports Register" on page 8-31   |
| Counter 2 Interval Time Status Byte Format<br>Counter 2 Counter Access Port Register          | 42h     | Section 8.3.2, "SBYTE_FMT---Interval Timer Status Byte Format Register" on page 8-30<br>Section 8.3.3, "Counter Access Ports Register" on page 8-31   |
| Timer Control Word Register<br>Timer Control Word Register Read Back<br>Counter Latch Command | 43h     | Section 8.3.1, "TCW---Timer Control Word Register" on page 8-28<br>Section 8.3.1.1, "RDBK_CMD---Read Back Command" on page 8-29<br>Section 8.3.1.2, "LTCH_CMD---Counter Latch Command" on page 8-30 |
| <b>Aliased at 40h–43h</b>   | 50h–53h |   |
| NMI Status and Control Register   | 61h     | Section 8.7.1, "NMI_SC---NMI Status and Control Register" on page 8-49  |
| NMI Enable Register   | 70h     | Section 8.7.2, "NMI_EN---NMI Enable (and Real Time Clock Index)" on page 8-50   |
| Real-Time Clock (Standard RAM) Index Register   | 70h     | Table 8-7 "RTC (Standard) RAM Bank" on page 8-45<br>Section 8.7.2, "NMI_EN---NMI Enable (and Real Time Clock Index)" on page 8-50   |
| Real-Time Clock (Standard RAM) Target Register  | 71h     | Table 8-7 "RTC (Standard) RAM Bank" on page 8-45  |
| Extended RAM Index Register   | 72h     |   |
| Extended RAM Target Register  | 73h     |   |
| <b>Aliased at 70h–71h</b>   | 74h–75h | Aliased if U128E bit in RTC Configuration Register is enabled<br>Section 8.1.23, "RTC_CONF---RTC Configuration Register (LPC I/F---D31:F0)" on page 8-14  |
| <b>Aliased at 72h–73h or 70h–71h</b>  | 76h–77h | Aliased to 70h–71h if U128E bit in RTC Configuration Register is enabled<br>Section 8.1.23, "RTC_CONF---RTC Configuration Register (LPC I/F---D31:F0)" on page 8-14                                 |
| Channel 2 DMA Memory Low Page Register  | 81h     | Section 8.2.3, "DMAMEM_LP---DMA Memory Low Page Registers" on page 8-23   |
| Channel 3 DMA Memory Low Page Register  | 82h     | Section 8.2.3, "DMAMEM_LP---DMA Memory Low Page Registers" on page 8-23   |
| Channel 1 DMA Memory Low Page Register  | 83h     | Section 8.2.3, "DMAMEM_LP---DMA Memory Low Page Registers" on page 8-23   |
| Reserved Page Registers   | 84h–86h |   |
| Channel 0 DMA Memory Low Page Register  | 87h     | Section 8.2.3, "DMAMEM_LP---DMA Memory Low Page Registers" on page 8-23   |
| Reserved Page Register  | 88h     |   |



Table A-2. ICH Fixed I/O Registers (Sheet 3 of 4)

| Register Name                                   | Port                    | Datasheet Section and Location   |
|---|-------------------------|--|
| Channel 6 DMA Memory Low Page Register          | 89h                     | Section 8.2.3, "DMAMEM_LP—DMA Memory Low Page Registers" on page 8-23                      |
| Channel 7 DMA Memory Low Page Register          | 8Ah                     | Section 8.2.3, "DMAMEM_LP—DMA Memory Low Page Registers" on page 8-23                      |
| Channel 5 DMA Memory Low Page Register          | 8Bh                     | Section 8.2.3, "DMAMEM_LP—DMA Memory Low Page Registers" on page 8-23                      |
| Reserved Page Registers                         | 8Ch–8Eh                 |  |
| Refresh Low Page Register                       | 8Fh                     |  |
| <b>Aliased at 81h–8Fh</b>                       | 91h–9Fh<br>(except 92h) |  |
| Fast A20 and INIT Register                      | 92h                     | Section 8.7.3, "PORT92—Fast A20 and Init Register" on page 8-50                            |
| Slave PIC ICW1 Init. Cmd Word 1 Register        | A0h                     | Section 8.4.2, "ICW1—Initialization Command Word 1 Register" on page 8-33                  |
| Slave PIC OCW2 Op Ctrl Word 2 Register          |                         | Section 8.4.8, "OCW2—Operational Control Word 2 Register" on page 8-36                     |
| Slave PIC OCW3 Op Ctrl Word 3 Register          |                         | Section 8.4.9, "OCW3—Operational Control Word 3 Register" on page 8-37                     |
| Slave PIC ICW2 Init. Cmd Word 2 Register        | A1                      | Section 8.4.3, "ICW2—Initialization Command Word 2 Register" on page 8-34                  |
| Slave PIC ICW3 Init. Cmd Word 3 Register        |                         | Section 8.4.5, "ICW3—Slave Controller Initialization Command Word 3 Register" on page 8-35 |
| Slave PIC ICW4 Init. Cmd Word 4 Register        |                         | Section 8.4.6, "ICW4—Initialization Command Word 4 Register" on page 8-35                  |
| Slave PIC OCW1 Op Ctrl Word 1 Register          |                         | Section 8.4.7, "OCW1—Operational Control Word 1 (Interrupt Mask) Register" on page 8-35    |
| <b>Aliased at A0h–A1h</b>                       | A4h–A5h                 |  |
| <b>Aliased at A0h–A1h</b>                       | A8h–A9h                 |  |
| <b>Aliased at A0h–A1h</b>                       | ACh–ADh                 |  |
| <b>Aliased at A0h–A1h</b>                       | B0h–B1h                 |  |
| Advanced Power Management Control Port Register | B2h                     | Section 8.8.2.1, "APM_CNT—Advanced Power Management Control Port Register" on page 8-56    |
| Advanced Power Management Status Port Register  | B3h                     | Section 8.8.2.2, "APM_STS—Advanced Power Management Status Port Register" on page 8-56     |
| <b>Aliased at A0h–A1h</b>                       | B4h–B5h                 |  |
| <b>Aliased at A0h–A1h</b>                       | B8h–B9h                 |  |
| <b>Aliased at A0h–A1h</b>                       | BCh–BDh                 |  |
| Channel 4 DMA Base & Current Address Register   | C0h                     | Section 8.2.1, "DMABASE_CA—DMA Base and Current Address Registers" on page 8-22            |
| <b>Aliased at C0h</b>                           | C1h                     |  |
| Channel 4 DMA Base & Current Count Register     | C2h                     | Section 8.2.2, "DMABASE_CC—DMA Base and Current Count Registers" on page 8-23              |
| <b>Aliased at C2h</b>                           | C3h                     |  |
| Channel 5 DMA Base & Current Address Register   | C4h                     | Section 8.2.1, "DMABASE_CA—DMA Base and Current Address Registers" on page 8-22            |
| <b>Aliased at C4h</b>                           | C5h                     |  |
| Channel 5 DMA Base & Current Count Register     | C6h                     | Section 8.2.2, "DMABASE_CC—DMA Base and Current Count Registers" on page 8-23              |
| <b>Aliased at C6h</b>                           | C7h                     |  |
| Channel 6 DMA Base & Current Address Register   | C8h                     | Section 8.2.1, "DMABASE_CA—DMA Base and Current Address Registers" on page 8-22            |

Table A-2. ICH Fixed I/O Registers (Sheet 4 of 4)

| Register Name   | Port      | Datasheet Section and Location  |
|---|-----------|---|
| <b>Aliased at C8h</b>   | C9h       |   |
| Channel 6 DMA Base & Current Count Register                         | CAh       | Section 8.2.2, "DMABASE_CC—DMA Base and Current Count Registers" on page 8-23   |
| <b>Aliased at CAh</b>   | CBh       |   |
| Channel 7 DMA Base & Current Address Register                       | CCh       | Section 8.2.1, "DMABASE_CA—DMA Base and Current Address Registers" on page 8-22                                       |
| <b>Aliased at CCh</b>   | CDh       |   |
| Channel 7 DMA Base & Current Count Register                         | CEh       | Section 8.2.2, "DMABASE_CC—DMA Base and Current Count Registers" on page 8-23   |
| <b>Aliased at CEh</b>   | CFh       |   |
| Channel 4-7 DMA Command Register<br>Channel 4-7 DMA Status Register | D0h       | Section 8.2.4, "DMACMD—DMA Command Register" on page 8-24<br>Section 8.2.5, "DMASTA—DMA Status Register" on page 8-24 |
| <b>Aliased at D0h</b>   | D1h       |   |
| Channel 4-7 DMA Write Single Mask Register                          | D4h       | Section 8.2.6, "DMA_WRSMSK—DMA Write Single Mask Register" on page 8-25   |
| <b>Aliased at D4h</b>   | D5h       |   |
| Channel 4-7 DMA Channel Mode Register                               | D6h       | Section 8.2.7, "DMACH_MODE—DMA Channel Mode Register" on page 8-25  |
| <b>Aliased at D6h</b>   | D7h       |   |
| Channel 4-7 DMA Clear Byte Pointer Register                         | D8h       | Section 8.2.8, "DMA Clear Byte Pointer Register" on page 8-26   |
| <b>Aliased at D8h</b>   | D9h       |   |
| Channel 4-7 DMA Master Clear Register                               | DAh       | Section 8.2.9, "DMA Master Clear Register" on page 8-26   |
| <b>Aliased at DAh</b>   | DBh       |   |
| Channel 4-7 DMA Clear Mask Register                                 | DCh       | Section 8.2.10, "DMA_CLMSK—DMA Clear Mask Register" on page 8-26  |
| <b>Aliased at DCh</b>   | DEh       |   |
| Channel 4-7 DMA Write All Mask Register                             | DEh       | Section 8.2.11, "DMA_WRMSK—DMA Write All Mask Register" on page 8-27  |
| <b>Aliased at DEh</b>   | DFh       |   |
| Coprocessor Error Register  | F0h       | Section 8.7.4, "COPROC_ERR—Coprocessor Error Register" on page 8-50   |
| PIO Mode Command Block Offset for Secondary Drive                   | 170h–177h | See ATA Specification for detailed register description   |
| PIO Mode Command Block Offset for Primary Drive                     | 1F0h–1F7h | See ATA Specification for detailed register description   |
| PIO Mode Control Block Offset for Secondary Drive                   | 376h      | See ATA Specification for detailed register description   |
| PIO Mode Control Block Offset for Primary Drive                     | 3F6h      | See ATA Specification for detailed register description   |
| Master PIC Edge/Level Triggered Register                            | 4D0h      | Section 8.4.10, "ELCR1—Master Controller Edge/Level Triggered Register" on page 8-38                                  |
| Slave PIC Edge/Level Triggered Register                             | 4D1h      | Section 8.4.11, "ELCR2—Slave Controller Edge/Level Triggered Register" on page 8-38                                   |
| Reset Control Register  | CF9h      | Section 8.7.5, "RST_CNT—Reset Control Register" on page 8-51  |

**NOTE:** When the POS\_DEC\_EN bit is set, additional I/O ports get positively decoded by the ICH. Refer to Section 8.1.24 through Section 8.1.29 for a listing of these ranges.

Table A-3. ICH Variable I/O Registers (Sheet 1 of 5)

| Register Name   | Offset  | Datasheet Section and Location   |
|---|---------|--|
| <b>Power Management I/O Registers at PMBASE+Offset</b><br>PMBASE set in Section 8.1.10, "PMBASE—ACPI Base Address (LPC I/F—D31.F0)" on page 8-6 |         |  |
| PM1 Status  | 00h–01h | Section 8.8.3.1, "PM1_STS—Power Management 1 Status Register" on page 8-58             |
| PM1 Enable  | 02h–03h | Section 8.8.3.2, "PM1_EN—Power Management 1 Enables Register" on page 8-59             |
| PM1 Control   | 04h–07h | Section 8.8.3.3, "PM1_CNT—Power Management 1 Control" on page 8-60                     |
| PM1 Timer   | 08h–0Bh | Section 8.8.3.4, "PM1_TMR—Power Management 1 Timer Register" on page 8-60              |
| Processor Control   | 10h–13h | Section 8.8.3.5, "PROC_CNT—Processor Control Register" on page 8-61                    |
| Level 2 Register  | 14h     | Section 8.8.3.6, "LV2 — Level 2 Register" on page 8-62                                 |
| General Purpose Event 0 Status  | 28h–29h | Section 8.8.3.7, "GPE0_STS—General Purpose Event 0 Status Register" on page 8-62       |
| General Purpose Event 0 Enables   | 2Ah–2Bh | Section 8.8.3.8, "GPE0_EN—General Purpose Event 0 Enables Register" on page 8-63       |
| General Purpose Event 1 Status  | 2Ch–2Dh | Section 8.8.3.9, "GPE1_STS—General Purpose Event 1 Status Register" on page 8-64       |
| General Purpose Event 1 Enables   | 2Eh–2Fh | Section 8.8.3.10, "GPE1_EN—General Purpose Event 1 Enable Register" on page 8-65       |
| SMI# Control and Enable   | 30h–31h | Section 8.8.3.11, "SMI_EN—SMI Control and Enable Register" on page 8-65                |
| SMI Status Register   | 34h–35h | Section 8.8.3.12, "SMI_STS—SMI Status Register" on page 8-66                           |
| Monitor SMI Status  | 40h     | Section 8.8.3.13, "IOMON_STS_EN — I/O Monitor Status and Enable Register" on page 8-67 |
| Device Activity Status  | 44h     | Section 8.8.3.14, "DEVACT_STS — Device Activity Status Register" on page 8-68          |
| Bus Address Tracker   | 4Ch     | Section 8.8.3.15, "BUS_ADDR_TRACK— Bus Address Tracker" on page 8-68                   |
| Bus Cycle Tracker   | 4Eh     | Section 8.8.3.16, "BUS_CYC_TRACK— Bus Cycle Tracker" on page 8-69                      |

Table A-3. ICH Variable I/O Registers (Sheet 2 of 5)

| Register Name  | Offset  | Datasheet Section and Location   |
|--|---------|--|
| <b>TCO I/O Registers at TCOBASE + Offset</b><br>TCOBASE = PMBASE + 40h<br>PMBASE is set in Section 3.1.10, "PMBASE—ACPI Base Address (LPC I/F—D31:F0)" on page 8-6 |         |  |
| TCO_RLD: TCO Timer Reload and Current Value  | 00h     | Section 8.9.2, "TCO1_RLD—TCO Timer Reload and Current Value" on page 8-70                |
| TCO_TMR: TCO Timer Initial Value   | 01h     | Section 8.9.3, "TCO1_TMR—TCO Timer initial Value" on page 8-70                           |
| TCO_DAT_IN: TCO Data In  | 02h     | Section 8.9.4, "TCO1_DAT_IN—TCO Data In Register" on page 8-71                           |
| TCO_DAT_OUT: TCO Data Out  | 03h     | Section 8.9.5, "TCO1_DAT_OUT—TCO Data Out Register" on page 8-71                         |
| TCO1_STS : TCO Status  | 04h–05h | Section 8.9.6, "TCO1_STS—TCO1 Status Register" on page 8-71                              |
| TCO2_STS : TCO Status  | 06h–07h | Section 8.9.7, "TCO2_STS—TCO2 Status Register" on page 8-72                              |
| TCO1_CNT: TCO Control  | 08h–09h | Section 8.9.8, "TCO1_CNT—TCO1 Control Register" on page 8-73                             |
| TCO2_CNT: TCO Control  | 0Ah–0Bh | Section 8.9.9, "TCO2_CNT—TCO2 Control Register" on page 8-73                             |
| <b>GPIO I/O Registers at GPIOBASE + Offset</b><br>GPIOBASE is set in Section 8.1.14, "GPIOBASE—GPIO Base Address (LPC I/F—D31:F0)" on page 8-8                     |         |  |
| GPIO Use Select  | 00h–03h | Section 8.10.2, "GPIO_USE_SEL—GPIO Use Select Register" on page 8-79                     |
| GPIO Input/Output Select   | 04h–07h | Section 8.10.3, "GP_IO_SEL—GPIO Input/Output Select Register" on page 8-80               |
| GPIO Level for Input or Output   | 0Ch–0Fh | Section 8.10.4, "GP_LVL—GPIO Level for Input or Output Register" on page 8-81            |
| GPIO TTL Select  | 14h–17h | Section 8.10.5, "GPO_TTL—GPIO TTL Select Register" on page 8-81                          |
| GPIO Blink Enable  | 18h–1Bh | Section 8.10.6, "GPO_BLINK—GPO Blink Enable Register" on page 8-82                       |
| GPIO Signal Invert   | 2Ch–2Fh | Section 8.10.7, "GPI_INV—GPIO Signal Invert Register" on page 8-82                       |
| <b>BMIDE I/O Registers at BM_BASE + Offset</b><br>BM_BASE is set at Section 9.1.11, "BM_BASE—Bus Master Base Address Register (IDE—D31:F1)" on page 9-5            |         |  |
| Command Register Primary   | 00h     | Section 9.2.1, "BMIC[P,S]—Bus Master IDE Command Register" on page 9-12                  |
| Status Register Primary  | 02h     | Section 9.2.2, "BMIS[P,S]—Bus Master IDE Status Register" on page 9-13                   |
| Descriptor Table Pointer Primary   | 04h–07h | Section 9.2.3, "BMID[P,S]—Bus Master IDE Descriptor Table Pointer Register" on page 9-14 |
| Command Register Secondary   | 08h     | Section 9.2.1, "BMIC[P,S]—Bus Master IDE Command Register" on page 9-12                  |
| Status Register Secondary  | 0Ah     | Section 9.2.2, "BMIS[P,S]—Bus Master IDE Status Register" on page 9-13                   |
| Descriptor Table Pointer Secondary   | 0Ch–0Fh | Section 9.2.3, "BMID[P,S]—Bus Master IDE Descriptor Table Pointer Register" on page 9-14 |

Table A-3. ICH Variable I/O Registers (Sheet 3 of 5)

| Register Name  | Offset  | Datasheet Section and Location  |
|--|---------|---|
| <b>USB I/O Registers at Base Address + Offset</b><br>USB Base Address is set at Section 10.1.10, "BASE---Base Address Register (USB---D31:F2)" on page 10-4                |         |   |
| USB Command Register   | 00h-01h | Section 10.2.1, "USBCMD---USB Command Register" on page 10-8                      |
| USB Status Register  | 02h-03h | Section 10.2.2, "USBSTA---USB Status Register" on page 10-11                      |
| USB Interrupt Enable   | 04h-05h | Section 10.2.3, "USBINTR---Interrupt Enable Register" on page 10-12               |
| USB Frame Number   | 06h-07h | Section 10.2.4, "FRNUM---Frame Number Register" on page 10-12                     |
| USB Frame List Base Address  | 08h-0Bh | Section 10.2.5, "FRBASEADD---Frame List Base Address" on page 10-13               |
| USB Start of Frame Modify  | 0Ch     | Section 10.2.6, "SOFMOD---Start of Frame Modify Register" on page 10-13           |
| Port 0 Status/Control  | 10h-11h | Section 10.2.7, "PORTSC[0,1]---Port Status and Control Register" on page 10-14    |
| Port 1 Status/Control  | 12h-13h | Section 10.2.7, "PORTSC[0,1]---Port Status and Control Register" on page 10-14    |
| Loop Back Test Data  | 18h     |   |
| <b>SMBus I/O Registers at SMB_BASE + Offset</b><br>SMB_BASE is set at Section 11.1.10, "SMB_BASE---SMBus Base Address Register (SMBUS---D31:F3)" on page 11-4              |         |   |
| Host Status  | 00h     | Section 11.2.1, "HST_STA---Host Status Register" on page 11-7                     |
| Host Control   | 02h     | Section 11.2.2, "HST_CNT---Host Control Register" on page 11-8                    |
| Host Command   | 03h     | Section 11.2.3, "HST_CMD---Host Command Register" on page 11-9                    |
| Transmit Slave Address   | 04h     | Section 11.2.4, "XMIT_SLVA---Transmit Slave Address Register" on page 11-9        |
| Host Data 0  | 05h     | Section 11.2.5, "D0---Data 0 Register" on page 11-9                               |
| Host Data 1  | 06h     | Section 11.2.6, "D1---Data 1 Register" on page 11-9                               |
| Block Data Byte  | 07h     | Section 11.2.7, "BLOCK_DB---Block Data Byte Register" on page 11-10               |
| <b>AC'97 Audio I/O Registers at NAMBAR + Offset</b><br>NAMBAR is set at Section 12.1.10, "NAMBAR---Native Audio Mixer Base Address Register (Audio---D31:F5)" on page 12-5 |         |   |
| PCM In Buffer Descriptor list Base Address Register  | 00h     | Section 12.2.1, "x_BDBAR---Buffer Descriptor Base Address Register" on page 12-10 |
| PCM In Current Index Value   | 04h     | Section 12.2.2, "x_CIV---Current Index Value Register" on page 12-10              |
| PCM In Last Valid Index  | 05h     | Section 12.2.3, "x_LVI---Last Valid Index Register" on page 12-10                 |
| PCM In Status Register   | 06h     | Section 12.2.4, "x_SR---Status Register" on page 12-11                            |

Table A-3. ICH Variable I/O Registers (Sheet 4 of 5)

| Register Name  | Offset | Datasheet Section and Location  |
|--|--------|---|
| PCM In Position In Current Buffer                    | 08h    | Section 12.2.5, "x_PICB—Position in Current Buffer Register" on page 12-12      |
| PCM In Prefetched Index Value                        | 0Ah    | Section 12.2.6, "x_PIV—Prefetched Index Value Register" on page 12-12           |
| PCM In Control Register                              | 0Bh    | Section 12.2.7, "x_CR—Control Register" on page 12-13                           |
| PCM Out Buffer Descriptor list Base Address Register | 10h    | Section 12.2.1, "x_BDBAR—Buffer Descriptor Base Address Register" on page 12-10 |
| PCM Out Current Index Value                          | 14h    | Section 12.2.2, "x_CIV—Current Index Value Register" on page 12-10              |
| PCM Out Last Valid Index                             | 15h    | Section 12.2.3, "x_LVI—Last Valid Index Register" on page 12-10                 |
| PCM Out Status Register                              | 16h    | Section 12.2.4, "x_SR—Status Register" on page 12-11                            |
| PCM Out Position In Current Buffer                   | 18h    | Section 12.2.5, "x_PICB—Position in Current Buffer Register" on page 12-12      |
| PCM Out Prefetched Index Value                       | 1Ah    | Section 12.2.6, "x_PIV—Prefetched Index Value Register" on page 12-12           |
| PCM Out Control Register                             | 1Bh    | Section 12.2.7, "x_CR—Control Register" on page 12-13                           |
| Mic. In Buffer Descriptor list Base Address Register | 20h    | Section 12.2.1, "x_BDBAR—Buffer Descriptor Base Address Register" on page 12-10 |
| Mic. In Current Index Value                          | 24h    | Section 12.2.2, "x_CIV—Current Index Value Register" on page 12-10              |
| Mic. In Last Valid Index                             | 25h    | Section 12.2.3, "x_LVI—Last Valid Index Register" on page 12-10                 |
| Mic. In Status Register                              | 26h    | Section 12.2.4, "x_SR—Status Register" on page 12-11                            |
| Mic In Position In Current Buffer                    | 28h    | Section 12.2.5, "x_PICB—Position in Current Buffer Register" on page 12-12      |
| Mic. In Prefetched Index Value                       | 2Ah    | Section 12.2.6, "x_PIV—Prefetched Index Value Register" on page 12-12           |
| Mic. In Control Register                             | 2Bh    | Section 12.2.7, "x_CR—Control Register" on page 12-13                           |
| Global Control                                       | 2Ch    | Section 12.2.8, "GLOB_CNT—Global Control Register" on page 12-14                |
| Global Status  | 30h    | Section 12.2.9, "GLOB_STA—Global Status Register" on page 12-14                 |
| Codec Access Semaphore Register                      | 34h    | Section 12.2.10, "CAS—Codec Access Semaphore Register" on page 12-15            |

Table A-3. ICH Variable I/O Registers (Sheet 5 of 5)

| Register Name  | Offset | Datasheet Section and Location   |
|--|--------|--|
| AC'97 Modem I/O Registers at MBAR + Offset<br>MBAR is set in Section 13.1.11, "MBAR—Modem Base Address Register (Modem—D31:F5)" on page 13-5 |        |  |
| Modem In Buffer Descriptor List Base Address Register  | 00h    | Section 13.2.1, "x_BDBAR—Buffer Descriptor List Base Address Register" on page 13-10 |
| Modem In Current Index Value Register  | 04h    | Section 13.2.2, "x_CIV—Current Index Value Register" on page 13-10                   |
| Modem In Last Valid Index Register   | 05h    | Section 13.2.3, "x_LVI—Last Valid Index Register" on page 13-10                      |
| Modem In Status Register   | 06h    | Section 13.2.4, "x_SR—Status Register" on page 13-11                                 |
| Modem In Position In Current Buffer Register   | 08h    | Section 13.2.5, "x_PICB—Position In Current Buffer Register" on page 13-12           |
| Modem In Prefetch Index Value Register   | 0Ah    | Section 13.2.6, "x_PIV—Prefetch Index Value Register" on page 13-12                  |
| Modem In Control Register  | 0Bh    | Section 13.2.7, "x_CR—Control Register" on page 13-13                                |
| Modem Out Buffer Descriptor List Base Address Register   | 10h    | Section 13.2.1, "x_BDBAR—Buffer Descriptor List Base Address Register" on page 13-10 |
| Modem Out Current Index Value Register   | 14h    | Section 13.2.2, "x_CIV—Current Index Value Register" on page 13-10                   |
| Modem Out Last Valid Register  | 15h    | Section 13.2.3, "x_LVI—Last Valid Index Register" on page 13-10                      |
| Modem Out Status Register  | 16h    | Section 13.2.4, "x_SR—Status Register" on page 13-11                                 |
| Modem In Position In Current Buffer Register   | 18h    | Section 13.2.5, "x_PICB—Position In Current Buffer Register" on page 13-12           |
| Modem Out Prefetched Index Register  | 1Ah    | Section 13.2.6, "x_PIV—Prefetch Index Value Register" on page 13-12                  |
| Modem Out Control Register   | 1Bh    | Section 13.2.7, "x_CR—Control Register" on page 13-13                                |
| Global Control   | 3Ch    | Section 13.2.8, "GLOB_CNT—Global Control Register" on page 13-14                     |
| Global Status  | 40h    | Section 13.2.9, "GLOB_STA—Global Status Register" on page 13-14                      |
| Codec Access Semaphore Register  | 44h    | Section 13.2.10, "CAS—Codec Access Semaphore Register" on page 13-16                 |

