FireStar/FireStar ACPI

64-Bit CPU Single Chip
Notebook Solution

Preliminary Data Book

Revision: 1.0
912-2000-015
February 28, 1997

OPTi Inc.

888 Tasman Drive
Milpitas, CA 95035
Tel: (408) 486-8000
Fax: (408) 486-8001
WWW: http://www.opti.com/

OPTi Inc.
888 Tasman Drive
Milpitas, CA 95035
(408) 486-8000
Fax: (408) 486-8001

Revision Document

Product Name: FireStar / FireStar ACPI (82C700) Single-Chip Notebook Solution

Document Title: Preliminary Data Book (#912-2000-015) Revision 1.0

Date: February 28, 1997

Scope

This document provides the revisions that have been incorporated in Revision 1.0 of the 82C700 Preliminary Data Book, #912-2000-015 since Revision 0.2 dated May 8, 1996.

The data book includes details regarding FireStar ACPI (hence the name change of the data book) and denotes, where appropriate, if the discussion is applicable only to FireStar ACPI.

Revision History (0.2 to 1.0)

Section 1.0 Features

- Figure 1-1 FireStar System Block Diagram
  - Designated PCI bus
  - Modified address line
  - Changed name of CPU (was designated as 64-Bit 586 CPU - FireStar supports both 586- and 686-class)
- Removed Unified Memory Interface (totally)
- Added ACPI Implementation bullet

Section 2.0 Overview

- No changes

Section 3.0 Signal Definitions

3.1 Terminology/Nomenclature Conventions

- No changes

3.2 Signal Descriptions

Summary the major changes made to the pin function assignments:

- Removed Boundary Scan Signal functions (all except TMS)
- Removed UMA interface functions
- Changed three of the VCC_CPU power pins to CPU_CORE (AB19, H22, K5) in preparation for future 2.5V CPU interface

Summary of the general format changes made:

- Expanded greatly on detailing of individual signal descriptions
- Designated in "Pin No." column of description tables which pins are used as strap option selection pins
- Clearly specified which PIO pins could be programmed as outputs only (PIO0 through PIO5) in description tables
- Adjusted pin diagram (Figure 3-1), alphabetical pin cross-reference list (Table 3-2) and the signal descriptions accordingly with the pin modifications listed in Table 1 (of this document

Table 1 (of this document) lists the actual pin modifications between data book 0.2 and 1.0 (alphabetically).

Table 1 FireStar Pin Modifications

Data Book Rev. 0.2 Signal Name	Pin No.	Data Book Rev. 1.0 Signal Name
BWE#+RAS4	P4	BWE#
CAS0#	B10	CAS0#+SDDQM0#
CAS1#	C10	CAS1#+SDDQM1#
CAS2#	D10	CAS2#+SDDQM2#
CAS3#	A11	CAS3#+SDDQM3#
CAS4#	B11	CAS4#+SDDQM4#
CAS5#	C11	CAS5#+SDDQM5#
CAS6#	D11	CAS6#+SDDQM6#
CAS7#	A12	CAS7#+SDDQM7#
CDOE#	P1	CDOE#+PIO0
CMD#+PCICLK3	AB20	CMD#+DIRTY+PCICLK3
DWE#	E10	DWE#+SDWE#
EADS#	T4	EADS#+WB/WT#
GNT3#+UMAGNT#	AC18	GNT3#
IRQSER+SOUT#	AE18	IRQSER+SDCKE+SOUT#
32KHZ	C7	OSC32
PWRGD+TRST#	H26	PWRGD
RAS0#	E12	RAS0#+SDCS0#
RAS1#+PIO5	E13	RAS1#+SDCS1#+PIO5
RAS2#+PIO4	B12	RAS2#+SDCS2#+PIO4
RAS3#+MA12	C12	RAS3#+SDCS3#+MA12
RSVD	E22	RAS4#+MA12
REQ3#+UMAREQ#	AD18	REQ3#
TCK	A8	SDCAS#
TMS	D7	SDRAS#
SEL#/ATB#+PIO14	AC20	SEL#/ATB#+SDCKE+PIO14
TAG0	E9	TAG0+CAS0#
TAG1+START#	D9	TAG1+CAS1#+START#
TAG2+START#	C9	TAG2+CAS2#+START#
TAG3+SBOFF#	B9	TAG3+CAS3#+SBOFF#
TAG4	A9	TAG4+CAS4#+SDCKE
TAG5	D8	TAG5+CAS5#+DWE#
TAG6	C8	TAG6+CAS6#+SDCAS#
TAG7	B8	TAG7+CAS7#+SDRAS#
TDI	B7	RSVD
TDO	A7	SDCKE (FS ACPI, RSVD in non-ACPI ver.)
RSVD	AB5	TMS
VCC_CPU	H22	VCC_CORE
VCC_CPU	K5	VCC_CORE
VCC_CPU	AB19	VCC_CORE
W/R#	AA5	W/R#+INV

3.3 Programmable I/O Pins

- Added note in opening text:
  "Any signal that can potentially be programmed as a PIO pin will function as a PIO pin only if the corresponding CPU register is programmed to a non-zero value."
- Table 3-3 PIO Functions
  - Misc. Outputs Group 3h
    Added functions: FAN, PCTLL, ATCLK/2
  - IDE Controller Outputs Group 4h
    Added functions: DDACK0-0#, DDACK0-1#, DDACK1-0#, DDACK1-1#
  - ACPI Inputs Group 7h - New group added to table
- Table 3-6 Register Programmable PIO Pins
  - Added corresponding pin names in register titles for easier reading

3.4 Strap Selected Options

- Table 3-7 Strap Options
  - Added corresponding pin number to table
  - Added BOFF# strap option
  - Combined RTCAS and A20M# into one strap option, RTCAS:A20M#
  - Added PCICLK0 strap option
  - Added RSVD strap option (preparation for 2.5V CPU interface future revision)
- Table 3-8 Strap Option Readback Registers
  - New table

Section 4.0 Functional Description

Major revisions to the functional description section include:

- Removed UMA Support (was Section 4.9)
- Added new section on ACPI Implementation (currently Section 4.16)
- Rearranged IDE Interface section for better flow and provided more details on emulated bus master mode
- Added SDRAM detection algorithm in DRAM Controller section
- Pulled all clock information into one section (System Clocks, Section 4.4)

Section 5.0 Register Descriptions

- Added the following notes to opening text:

Notes: 1. Bits/registers that are new or have changed (between Data Book Rev 0.2 and Data Book Rev 1.0), are underlined and denoted with a change bar in the margin.

2. All bits/registers are read/write and their default value is 0 unless otherwise specified.

3. All reserved bits/registers **MUST** be written to 0 unless otherwise specified.

In this portion of the revision document, the tables that correspond to the register tables in the data book will specify either:

a. No change,
b. reference only the bits/registers that are new or changed, and if any
c. default changes

5.1 SYSCFG Register Space

- No changes to opening text.

Table 5-5 SYSCFG 00h-FFh

Loc.	Register Name	Revision
00h	Byte Merge/Prefetch & Sony Cache Module Control Register	No change
01h	DRAM Control Register 1	No change
02h	Cache Control Register 1	Bits [7,6]
03h	Cache Control Register 2	No change
04h	Shadow RAM Control Register 1	Added text to footnotes 2 and 3
05h	Shadow RAM Control Register 2	No change
06h	Shadow RAM Control Register 3	No change
07h	Tag Test Register	No change
08h	CPU Cache Control Register	Bits [4,2]
09h	System Memory Function Register	No change
0Ah	DRAM Hole A Address Decode Register	No change
0Bh	DRAM Hole B Address Decode Register	No change
0Ch	DRAM Hole Higher Address	Bits [7,5,4]
0Dh	Clock Control Register	Bits [7:4]
0Eh	PCI Master Burst Control Register 1	Bits [7:4]
0Fh	PCI Master Burst Control Register 2	Bits [7,5:3,1,0]
10h	Miscellaneous Control Register 1	No change
11h	Miscellaneous Control Register 2	Bits [4,3,1]
12h	Refresh Control Register	Bits [7,0]
13h	Memory Decode Control Register 1	No change
14h	Memory Decode Control Register 2	No change
15h	PCI Cycle Control Register 1	No change
16h	Dirty/Tag RAM Control Register	Bits [7,5], new footnote, default changed
17h	PCI Cycle Control Register 2	Bit [5]
18h	Interface Control Register	Bit [6]
	FS ACPI	Bits [7,3,1]
19h	Memory Decode Control Register 3	Bits [3:0]
1Ah	Memory Shadow Control Register 1	No change
1Bh	Memory Shadow Control Register 2	No change
1Ch	EDO DRAM Control Register	No change
1Dh	Miscellaneous Control Register 3	Bits [6,2]
1Eh	Control Register	Bits [6,4,2]
1Fh	EDO Timing Control Register	Bits [5,2,1]
20h	DRAM Burst Control Register	Bits [7,5]
21h	PCI Concurrency Control Register	Bit [0], default changed
22h	Inquire Cycle Control Register	Bits [7,5]
23h	Pre-Snoop Control Register	Bits [6,3:1]
24h	Asymmetric DRAM Configuration Register	No change
25h	GUI Memory Location Register	Bits [7:3,2,0]
26h	UMA Control Register 1	Bits [7,5,4,2:0]
27h	Miscellaneous Control Register 4	Bits [6,5,3,2:0]
28h	SDRAM Control Register 1	Bits [7:0]
29h	SDRAM Control Register 2	Bits [7:0]
2Ah	PCI-to-DRAM Control Register 1	Bits [7:0]
2Bh	PCI-to-DRAM Control Register 2	Bits [7:0]
2Ch	CPU-to-DRAM Buffer Control Register	Bits [6,5,3,1], new footnote
2Dh	Miscellaneous Control Register 5	Added text to footnote
2Eh	UMA Control Register 2	Bits [7:4,1,0]
2Fh	UMA Control Register 3	Bits [7:0], footnote
30h-37h	Reserved	No change
38h	NMI Trap Enable Register 1	No change
39h	NMI Trap Enable Register 2	No change
3Ah	NMI Trap Enable Register 3	No change
3Bh	NMI Trap Enable Register 4	No change
3Ch	NMI Trap Enable Register 5	No change
3Dh-3Fh	Reserved	No change

Table 5-5 SYSCFG 00h-FFh (cont.)

Loc.	Register Name	Revision
40h	PMU Control Register 1	Bit [7]
41h	DOZE_TIMER Register	No change
42h	If AEh[7] = 0: Clock Source Register 1	Added parameter "If AEh[7] = 0"
	If AEh[7] = 1: Clock Source Register 1A	New bits available "If AEh[7] = 1"
43h	PMU Control Register 2	No change
44h	LCD_TIMER Register	No change
45h	DSK_TIMER Register	No change
46h	KBD_TIMER Register	No change
47h	If AEh[7] = 0: GNR1_TIMER Register	Added parameter "If AEh[7] = 0"
	If AEh[7] = 1: GNR5_TIMER Register	New bits available "If AEh[7] = 1"
48h	If AEh[7] = 0: GNR1 Base Address Register	Added parameter "If AEh[7] = 0"
	If AEh[7] = 1: GNR5_Timer Base Address Register	New bits available "If AEh[7] = 1"
49h	If AEh[7] = 0: GNR1 Control Register	Added parameter "If AEh[7] = 0"
	If AEh[7] = 1: GNR5_Timer Control Register	New bits available "If AEh[7] = 1"
4Ah	Chip Select 0 Base Address Register	No change
4Bh	Chip Select 0 Control Register	No change
4Ch	Chip Select 1 Base Address Register	No change
4Dh	Chip Select 1 Control Register	No change
4Eh	If AEh[7] = 0: Idle Reload Event Enable Register 1	Added parameter "If AEh[7] = 0"
	If AEh[7] = 1: Idle Reload Event Enable Register 1A	New bits available "If AEh[7] = 1"
4Fh	IDLE_TIMER Register	No change
50h	PMU Control Register 3	No change
51h	Beeper Control Register	No change
52h	Scratchpad Register 1	No change
53h	Scratchpad Register 2	No change
54h	Power Control Latch Register 1	No change
55h	Power Control Latch Register 2	No change
56h	Reserved	No change
57h	PMU Control Register 4	Bits [3,2], default changed
58h	PMU Event Register 1	No change
59h	PMU Event Register 2	No change

Loc.	Register Name	Revision
5Ah	If AEh[7] = 0: PMU Event Register 3	Added parameter "If AEh[7] = 0"
	If AEh[7] = 1: PMU Event Register 3A	New bits available "If AEh[7] = 1"
5Bh	If AEh[7] = 0: PMU Event Register 4	Added parameter "If AEh[7] = 0"
	If AEh[7] = 1: PMU Event Register 4A	New bits available "If AEh[7] = 1"
5Ch	PMI SMI Source Register 1	No change
5Dh	If AEh[7] = 0: PMI SMI Source Register 2	No change
	If AEh[7] = 1: PMI SMI Source Register 2A	Bit [3]
5Eh	Reserved	No change
5Fh	PMU Control Register 5	No change
60h	R_Timer Count Register	No change
61h	Debounce Register	No change
62h	IRQ Doze Register 1	No change
63h	Idle Time-Out Select Register 1	No change
64h	INTRGRP IRQ Select Register 1	No change
65h	Doze Register	No change
66h	PMU Control Register 6	Bits [3:1]
67h	PMU Control Register 7	No change
68h	Clock Source Register 2	No change
69h	R_TIMER Register	No change
6Ah	RSMGRP IRQ Register 1	No change
6Bh	Resume Source Register	Bits [6,5]
6Ch	Scratchpad Register 3	No change
6Dh	Scratchpad Register 4	No change
6Eh	Scratchpad Register 5	No change
6Fh	Scratchpad Register 6	No change
70h	GNR1 Base Address Register 1	No change
71h	GNR1 Control Register 1	No change
72h	GNR1 Control Register 2	No change
73h	GNR2 Base Address Register 1	No change
74h	GNR2 Control Register 1	No change
75h	GNR2 Control Register 2	No change
76h	If AEh[7] = 0: Doze Reload Select Register 1	Added parameter "If AEh[7] = 0"
	If AEh[7] = 1: Doze Reload Select Register 1A	New bits available "If AEh[7] = 1"
77h	Doze Reload Select Register 2	No change
78h	Doze Reload Select Register 3	No change
79h	PMU Control Register 8	Bits [2,1]
7Ah	GNR3 Base Address Register 1	No change
7Bh	GNR3 Control Register 1	No change

Table 5-5 SYSCFG 00h-FFh (cont.)

Loc.	Register Name	Revision
7Ch	GNR3 Control Register 2	No change
7Dh	GNR4 Base Address Register 1	No change
7Eh	GNR4 Control Register 1	No change
7Fh	GNR4 Control Register 2	No change
80h	ICW1 Shadow Register for INTC1	No change
81h	ICW2 Shadow Register for INTC1	No change
82h	ICW3 Shadow Register for INTC1	No change
83h	ICW4 Shadow Register for INTC1	No change
84h	DMA In-Progress Register	No change
85h	OCW2 Shadow Register for INTC1	No change
86h	OCW3 Shadow Register for INTC1	No change
87h	Reserved	No change
88h	ICW1 Shadow Register for INTC2	No change
89h	ICW2 Shadow Register for INTC2	No change
8Ah	ICW3 Shadow Register for INTC2	No change
8Bh	ICW4 Shadow Register for INCT2	No change
8Ch	Reserved	No change
8Dh	OCW2 Shadow Register for INTC2	No change
8Eh	OCW3 Shadow Register for INTC2	No change
8Fh	Reserved	No change
90h	Timer Channel 0 Low Byte Register: A[7:0]	No change
91h	Timer Channel 0 High Byte Register: A[15:8]	No change
92h	Timer Channel 1 Low Byte Register: A[7:0]	No change
93h	Timer Channel 1 High Byte Register: A[15:8]	No change
94h	Timer Channel 2 Low Byte Register: A[7:0]	No change
95h	Timer Channel 2 High Byte Register: A[15:8]	No change
96h	Write Counter High/Low Byte Latch	No change
97h	Reserved	No change
98h	RTC Index Shadow Register	No change

Loc.	Register Name	Revision
99h	Interrupt Request Register for INCT1	No change
9Ah	Interrupt Request Register for INCT2	No change
9Bh	3F2h + 3F7h Shadow Register	No change
9Ch	372h + 377h Shadow Register	No change
9Dh-9Eh	Reserved	No change
9Fh	Port 064h Shadow Register	No change
A0h	Feature Control Register 1	No change
A1h	Feature Control Register 2	No change
A2h	If AEh[7] = 0: IRQ Doze Register 2	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1: IRQ Doze Register 2A	New bits available “If AEh[7] = 1”
A3h	Idle Time-Out Select Register 2	No change
A4h	INTRGRP IRQ Select Register 2	No change
A5h	Thermal Management Register 1	No change
A6h	Thermal Management Register 2	No change
A7h	Thermal Management Register 3	No change
A8h	Thermal Management Register 4	No change
A9h	Thermal Management Register 5	No change
AAh	Thermal Management Register 6	No change
ABh	Power Control Latch Register 3	No change
ACh	Reserved	No change
ADh	Feature Control Register 3	No change
AEh	GNR_ACCESS Feature Register 1	No change
AFh-B0h	Reserved	No change
B1h	RSMGRP IRQ Register 2	No change
B2h	If AEh[7] = 0: Clock Source Register 3	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1: Clock Source Register 3A	New bits available “If AEh[7] = 1”
B3h	Chip Select Cycle Type Register	No change
B4h	HDU_TIMER Register	No change
B5h	COM1_TIMER Register	No change
B6h	COM2_TIMER Register	No change
B7h	If AEh[7] = 0: GNR2_TIMER Register	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1: GNR6_TIMER Register	New bits available “If AEh[7] = 1”
B8h	If AEh[7] = 0: GNR2 Base Address Register	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1: GNR6 Base Address Register	New bits available “If AEh[7] = 1”

Table 5-5 SYSCFG 00h-FFh (cont.)

Loc.	Register Name	Revision
B9h	If AEh[7] = 0: GNR2 Control Register	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1: GNR6 Control Register	New bits available “If AEh[7] = 1”
BAh	Chip Select 2 Base Address Register	No change
BBh	Chip Select 2 Control Register	No change
BCh	Chip Select 3 Base Address Register	No change
BDh	Chip Select 3 Control Register	No change
BEh	If AEh[7] = 0: Idle Reload Event Enable Register 2	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1: Idle Reload Event Enable Register 2A	New bits available “If AEh[7] = 1”
BFh	Chip Select Granularity Register	No change
C0h-D4h	Reserved	No change
D5h	X Bus Positive Decode Register	Bits [7:0], new footnote
D6h	PMU Control Register 9	No change
D7h	Access Port Address Register 1	No change
D8h	If AEh[7] = 0: PMU Event Register 5	Added parameter “If AEh[7] = 0” and bits [7:0]
	If AEh[7] = 1: PMU Event Register 5A	New bits available “If AEh[7] = 1”
D9h	PMU Event Register 6	No change
DAh	Power Management Event Status Register	No change
DBh	If AEh[7] = 0: Next Access Event Generation Register 1	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1: Next Access Event Generation Register 1A	New bits available “If AEh[7] = 1”
DCh	If AEh[7] = 0: PMU SMI Source Register 1	No change
	If AEh[7] = 1: PMU SMI Source Register 1A	Bit 0
DDh	PMU SMI Source Register 2	No changed
	FS ACPI	Bit 7
DEh	If AEh[7] = 0: Current Access Event Generation Register 1	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1: Current Access Event Generation Register 1A	New bits available “If AEh[7] = 1”
DFh	If AEh[7] = 0:Activity Tracking Register 1	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1:Activity Tracking Register 1A	New bits available “If AEh[7] = 1”

Loc.	Register Name	Revision
E0h	If AEh[7] = 0: Activity Tracking Register 2	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1: Activity Tracking Register 2A	New bits available “If AEh[7] = 1”
E1h	If AEh[7] = 0: GNR3 Base Address Register	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1: GNR7 Base Address Register	New bits available “If AEh[7] = 1”
E2h	If AEh[7] = 0: GNR3 Control Register	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1: GNR7 Control Register	New bits available “If AEh[7] = 1”
E3h	If AEh[7] = 0: GNR4 Base Address Register	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1: GNR8 Base Address Register	New bits available “If AEh[7] = 1”
E4h	If AEh[7] = 0: GNR4 Control Register	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1: GNR8 Control Register	New bits available “If AEh[7] = 1”
E5h	GNR_ACCESS Feature Register 2	Bit [6]
E6h	If AEh[7] = 0: Clock Source Register 4	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1: Clock Source Register 4A	New bits available “If AEh[7] = 1”
E7h	If AEh[7] = 0: GNR3_TIMER Register	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1: GNR7_TIMER Register	New bits available “If AEh[7] = 1”
E8h	If AEh[7] = 0: GNR4_TIMER Register	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1: GNR8_TIMER Register	New bits available “If AEh[7] = 1”
E9h	If AEh[7] = 0: PMU Event Register 7	Added parameter “If AEh[7] = 0”
	If AEh[7] = 1: PMU Event Register 7A	New bits available “If AEh[7] = 1”
EAh	If AEh[7] = 0: PMU SMI Source Register 3	No change
	If AEh[7] = 1: PMU SMI Source Register 3A	Bits [1,0]
EBh	Access Port Address Register 2	No change
ECh	Write Trap Register 1	No change
EDh	Write Trap Register 2	No change
EEh	Power Control Latch Register 4	No change
EFh	Hot Docking Control Register 1	No change
F0h	Hot Docking Control Register 2	No change
F1h	Low Order Start Address for ROM Window	No change

Table 5-5 SYSCFG 00h-FFh (cont.)

Loc.	Register Name	Revision
F2h	High Order Start Address for ROM Window	No change
F3h	Thermal Management Register 7	No change
F4h	Thermal Management Register 8	No change
F5h	PMU Event Register 8	No change
	FS ACPI	Bits [7,6]
F6h	DMA Doze Reload Register 1	No change
F7h	DMA Doze Reload Register 2	No change
F8h	Compact ISA Control Register 1	No change

Loc.	Register Name	Revision
F9h	Compact ISA Control Register 2	No change
FAh	Compact ISA Control Register 3	Bits [7,6]
FBh	DMA Idle Reload Register	No change
FCh	IDE Power Management Assignment Register 1	Bits [7,6,3,2]
FDh	IDE Power Management Assignment Register 2	Bits [7,6,3,2]
FEh	GPCS# Global Control Register	No change
FFh	Reserved	No change

5.2 PCIDV0 Register Space

- Modified opening text slightly, removed reference to 82C701.

Table 5-6 PCIDV0 00h-FFh

Loc.	Register Name	Revision
00h	Vendor Identification Register - Byte 0	No change
01h	Vendor Identification Register - Byte 1	No change
02h	Device Identification Register - Byte 0	No change
03h	Device Identification Register - Byte 1	No change
04h	Command Register - Byte 0	Changed reference from 82C701 to 82C700 in bits [3:0]
05h	Command Register - Byte 1	No change
06h	Status Register - Byte 0	No change
07h	Status Register - Byte 1	Changed reference from 82C701 to 82C700 in bits [3:2]
08h	Revision Identification Register	Default changed
09h	Class Code Register - Byte 0	No change
0Ah	Class Code Register - Byte 1	No change
0Bh	Class Code Register - Byte 2	No change
0Ch	Reserved	No change
0Dh	Master Latency Timer Register	No change
0Eh	Header Type Register	No change
0Fh	Built-In Self-Test (BIST) Register	No change
10h-2Bh	Reserved	No change

Loc.	Register Name	Revision
2Ch-2Dh	Subsystem Vendor ID	New register
2Eh-2Fh	Subsystem ID	New register
30h-3Fh	Reserved	No change
40h	Memory Control Register - Byte 0	Bit [5]
41h	Memory Control Register - Byte 1	No change
42h	Memory Control Register - Byte 2	New register
43h	Internal Project Revision - Reserved	New register
44h	Data Path Register 1	Bits [6:5]
45h	Data Path Control Register 2	Bits [5,4,2,0]
46h	Data Path Control Register 3	No change
47h	Data Path Control Register 4	Bits [6:3], new footnote
48h	Data Path Control Register 5	New register
49h-4Bh	Reserved	No change
4Ch	MCACHE Control Register	New register
4Dh	Delay Adjustment Register	New register
4Eh	SDRAM Control Register	New register
4Fh-FFh	Reserved	No change

5.3 PCIDV1 Register Space

- Modified opening text slightly
- Added additional table (PCIDV1 table is now 5-8)

Table 5-8 PCIDV1 00h-FFh

Loc.	Register Name	Revision
00h	Vendor Identification Register - Byte 0	No change
01h	Vendor Identification Register - Byte 1	No change
02h	Device Identification Register - Byte 0	No change
03h	Device Identification Register - Byte 1	No change
04h	Command Register - Byte 0	No change
05h	Command Register - Byte 1	No change
06h	Status Register - Byte 0	No change
07h	Status Register - Byte 1	No change
08h	Revision Identification Register	Default changed
09h	Class Code Register - Byte 0	No change
0Ah	Class Code Register - Byte 1	No change
0Bh	Class Code Register - Byte 2	No change
0Ch	Reserved	No change
0Dh	Master Latency Timer Register	No change
0Eh	Header Type Register	No change
0Fh	Built-In Self-Test (BIST) Register	No change
10h-2Bh	Reserved	No change
2Ch-2Dh	Subsystem Vendor ID	New register
2Eh-2Fh	Subsystem ID	New register
30h-40h	Reserved	No change
41h	Keyboard Controller Select Register	Bit [3]
42h	Reserved	No change
43h	Feature Control Register	No change
44h-45h	Reserved	No change
46h	PCI Control Register B - Byte 0	No change
47h	PCI Control Register B - Byte 1	No change
48h	Strap Option Readback Register - Byte 0	No change
49h	Strap Option Readback Register - Byte 1	Bits [4:0], new footnote
4Ah	ROM Chip Select Register 1	No change
4Bh	ROM Chip Select Register 2	No change
4Ch-4Dh	Reserved	No change
4Eh	Miscellaneous Control Register 1	Bits [7:4]
	FS ACPI	Bits [5,4]
4Fh	Miscellaneous Control Register 2	Bit [7,2], new footnote
	FS ACPI	Bit [7]
50h-51h	Reserved	No change
52h	Miscellaneous Controller Register 3	Bit [2]
	FS ACPI	Bits [4,3]
53h	Miscellaneous Controller Register 4	Bits [7:1]
54h	IRQ Driveback Address Register - Byte 0: Address Bits [7:0]	No change
55h	IRQ Driveback Address Register - Byte 1: Address Bits [15:8]	No change
56h	IRQ Driveback Address Register - Byte 2: Address Bits [23:16]	No change
57h	IRQ Driveback Address Register - Byte 3: Address Bits [31:24]	No change
58h	DRQ Remap Base Address Register - Byte 0: Address Bits [7:0]	No change
59h	DRQ Remap Base Address Register - Byte 1: Address Bits [15:8]	No change
5Ah	DRQ Remap Base Address Register - Byte 2: Address Bits [23:16]	No change
5Bh	DRQ Remap Base Address Register - Byte 3: Address Bits [31:24]	No change
5Ch	DMA Channel Selector Register	No change
5Dh	Reserved	No change
5Eh	IRQ Scheme Management Register	No change
5Fh	SYSCFG Base Select Register	No change
60h	IRQ Driveback Data Register - Byte 0: Data Bits [7:0]	No change
61h	IRQ Driveback Data Register - Byte 1: Data Bits [15:8]	No change
62h	IRQ Driveback Data Register - Byte 2: Data Bits [23:16]	No change
63h	IRQ Driveback Data Register - Byte 3: Data Bits [31:24]	No change
64h	PCI Master Control Register 1	No change
	FS ACPI	Bit [1]
65h	PCI Master Control Register 2	No change
	FS ACPI	Bit [6]
66h	Reserved	No change
67h	Miscellaneous Control Register 5	Bits [1,0]
68h	PCICLK Control Register 1	No change

Table 5-8 PCIDV1 00h-FFh (cont.)

Loc.	Register Name	Revision
69h	PCICLK Control Register 2	No change
6Ah	PCICLK Skew Adjust Register for PCICLK 0, 1, 2	No change
6Bh	PCICLK Skew Adjust Register for PCICLK 3, 4, 5	No change
6Ch-6Fh	Reserved	No change
70h	Leakage Control Register - Byte 0	Bits [1:0]
71h	Leakage Control Register - Byte 1	Bits [7:0]
72h	Leakage Control Register - Byte 2	Bits [7:4,1]
73h	Leakage Control Register - Byte 3	Bits [5:4]
74h	Leakage Control Register - Byte 4	Bits [3:0]
75h	Leakage Control Register - Byte 5	Bits [7:2]
	FS ACPI	Bits [3:2]
76h	Hot Docking Leakage Control Register	New register
77h-7Fh	Reserved	No change
80h	PIO0 Pin (CDOE#) Function Register	No change
81h	PIO1 Pin (TAGWE#) Function Register	No change
82h	PIO2 Pin (ADSC#) Function Register	No change
83h	PIO3 Pin (ADV#) Function Register	No change
84h	PIO4 Pin (RAS2#) Function Register	No change
85h	PIO5 Pin (RAS1#) Function Register	No change
86h	PIO6 Pin (CLKRUN#) Function Register	No change
87h	PIO7 Pin (REQ1#) Function Register	No change
88h	PIO8 Pin (REQ2#) Function Register	No change
89h	PIO9 Pin (DDRQ0) Function Register	No change
8Ah	PIO10 Pin (IRQ1) Function Register	No change
8Bh	PIO11 Pin (IRQ8#) Function Register	No change
8Ch	PIO12 Pin (IRQ12) Function Register	No change
8Dh	PIO13 Pin (IRQ14) Function Register	No change
8Eh	PIO14 Pin (SEL#/ATB#) Function Register	No change
8Fh	PIO15 Pin (RSTDRV) Function Register	No change
90h	PIO16 Pin (SA16) Function Register	No change
91h	PIO17 Pin (SA17) Function Register	No change
92h	PIO18 Pin (IO16#) Function Register	No change
93h	PIO19 Pin (M16#) Function Register	No change
94h	PIO20 Pin (SBHE#) Function Register	No change
95h	PIO21 Pin (SMRD#) Function Register	No change
96h	PIO22 Pin (SMWR#) Function Register	No change
97h	PIO23 Pin (ROMCS#) Function Register	No change

Loc.	Register Name	Revision
98h	PIO24 Pin (KBDCS#) Function Register	No change
99h	PIO25 Pin (DRQA) Function Register	No change
9Ah	PIO26 Pin (DRQB) Function Register	No change
9Bh	PIO27 Pin (DRQC) Function Register	No change
9Ch	PIO28 Pin (DRQD) Function Register	No change
9Dh	PIO29 Pin (DRQE) Function Register	No change
9Eh	PIO30 Pin (DRQF) Function Register	No change
9Fh	PIO31 Pin (DRQG) Function Register	No change
A0h	Logic Matrix Register 1	No change
A1h	Logic Matrix Register 2	No change
A2h	Logic Matrix Register 3	No change
A3h	Logic Matrix Register 4	No change
A4h	Logic Matrix Register 5	No change
A5h	Logic Matrix Register 6	No change
A6h	Logic Matrix Register 7	No change
A7h	Logic Matrix Register 8	No change
A8h	PIO Pin Current State Register 1	No change
A9h	PIO Pin Current State Register 2	No change
AAh	PIO Pin Current State Register 3	No change
ABh	PIO Pin Current State Register 4	No change
ACh-ADh	Reserved	No change
AEh	DBE# Select Register 1	No change
AFh	DBE# Select Register 2	No change
B0h	IRQA Interrupt Selection Register	No change
B1h	IRQB Interrupt Selection Register	No change
B2h	IRQC Interrupt Selection Register	No change
B3h	IRQD Interrupt Selection Register	No change
B4h	IRQE Interrupt Selection Register	No change
B5h	IRQF Interrupt Selection Register	No change
B6h	IRQG Interrupt Selection Register	No change
B7h	IRQH Interrupt Selection Register	No change
B8h	PCI Interrupt Selection Register 1	No change
B9h	PCI Interrupt Selection Register 2	No change
BAh	Serial IRQ Control Register 1	No change
BBh	Serial IRQ Control Register 2	No change
BCh-BFh	Reserved	No change
C0h	DMA Channels A and B Selection Register	No change
C1h	DMA Channels C and D Selection Register	No change
C2h	DMA Channel E Selection Register	No change
C3h	DMA Channels F and G Selection Register	No change

Table 5-8 PCIDV1 00h-FFh (cont.)

Loc.	Register Name	Revision
C4h-CFh	Reserved	No change
Note:	PCIDV1 D0h through EEh pertain only to FS ACPI Version. Otherwise they are reserved.	
D0h	FS ACPI: PM1_BLK Base Address Register - Byte 0: Address Bits [7:0]	New register
D1h	FS ACPI: PM1_BLK Base Address Register - Byte 1: Address Bits [15:8]	New register
D2h	FS ACPI: PM2_BLK Base Address Register - Byte 0: Address Bits [7:0]	New register
D3h	FS ACPI: PM2_BLK Base Address Register -Byte 1: Address Bits [15:8]	New register
D4h	FS ACPI: P_BLK Base Address Register - Byte 0: Address Bits [7:0]	New register
D5h	FS ACPI: P_BLK Base Address Register - Byte 1: Address Bits [15:8]	New register
D6h	FS ACPI: GPE0_BLK Base Address Register - Byte 0: Address Bits [7:0]	New register
D7h	FS ACPI: GPE0_BLK Base Address Register - Byte 0: Address Bits [15:8]	New register
D8h	FS ACPI: ACPI Source Control Register - Byte 0	New register
D9h	FS ACPI: ACPI Source Control Register - Byte 1	New register
DAh	FS ACPI: ACPI Source Status Register - Byte 0	New register
DBh	FS ACPI: ACPI Source Status Register - Byte 1	New register
DCh	FS ACPI: ACPI Event Resume Control Register - Byte 0	New register
DDh	FS ACPI: ACPI Event Resume Control Register - Byte 1	New register
DEh-DFh	Reserved	New register

Loc.	Register Name	Revision
E0h	FS ACPI: SLP_TYP Control Register - Byte 0	New register
E1h	FS ACPI: SLP_TYP Control Register - Byte 1	New register
E2h	FS ACPI: SLP_TYP Control Register - Byte 2	New register
E3h	FS ACPI: SLP_TYP Control Register - Byte 3	New register
E4h	FS ACPI: SLP_TYP Control Register - Byte 4	New register
E5h	FS ACPI: SLP_TYP Control Register - Byte 5	New register
E6h	FS ACPI: SLP_TYP Control Register - Byte 6	New register
E7h	FS ACPI: SLP_TYP Control Register - Byte 7	New register
E8h	FS ACPI: Power Control Latch Set Register	New register
E9h	Reserved	New register
EAh	FS ACPI: Power Control Readback Register - Byte 0	New register
EBh	FS ACPI: Power Control Readback Register - Byte 1	New register
ECh	FS ACPI: Power Control Readback Register - Byte 2	New register
EDh	FS ACPI: Power Control Readback Register - Byte 3	New register
EEh	FS ACPI: ACPI Thermal Control Register	New register
EFh-FDh	Reserved	No change
FEh	Stop Grant Cycle Generation Register	No change
FFh	Parity Error Cycle Generation Register	No change

5.4 IDE Register Space

5.4.1 IDE Configuration Registers

- No change in opening text

Table 5-9 PCIIDE 00h-47h

Loc	Register Name	Revision
00h	Vendor ID Register - Byte 0	No change
01h	Vendor ID Register - Byte 1	No change
02h	Device ID Register - Byte 0	No change
03h	Device ID Register - Byte 1	No change
04h	Command Register - Byte 0	No change
05h	Command Register - Byte 1	No change
06h	Status Register - Byte 0	No change
07h	Status Register - Byte 1	No change
08h	Revision ID Register	No change
09h	Class Code Register - Byte 0	No change
0Ah	Class Code Register - Byte 1	No change
0Bh	Class Code Register - Byte 2	No change
0Ch-0Dh	Reserved	No change
0Eh	Header Type Register	No change
0Fh	Built-In Self-Test Register	No change
10h-13h	Primary IDE Command Block Base Address Register	Clarified default
14h-17h	Primary IDE Control Block Base Address Register	Clarified default
18h-1Bh	Secondary IDE Command Block Base Address Register	Clarified default
1Ch-1Fh	Secondary IDE Control Block Base Address Register	Clarified default
20h-23h	Bus Master IDE Base Address Register	Default changed
24h-2Bh	Reserved	No change
2Ch-2Dh	Subsystem Vendor ID	New register
2Eh-2Fh	Subsystem ID	New register
30h-3Ah	Reserved	No change
3Ch	Interrupt Line Register	Default changed
3Dh	Interrupt Pin Register	No change
3Eh-3Fh	Reserved	No change
40h	IDE Initialization Control Register	No change
41h	Reserved	No change
42h	IDE Enhanced Feature Register	No change
	FS ACPI: IDE Enhanced Feature Register	Bit [6]
43h	IDE Enhanced Mode Register	No change
Note: The registers differ for FireStar and FS ACPI from 44h.		
FireStar		
44h	Emulated Bus Master Register	No change
45h	IDE Interrupt Selection Register	No change
FireStar ACPI		
44h	FS ACPI: Ultra DMA Configuration Register 1	New register
45h	FS ACPI: Ultra DMA Configuration Register 2	New register
46h	FS ACPI: Emulated IDE Configuration Register	New location
47h	FS ACPI: IDE Interrupt Selection Register	New location

5.4.2 IDE I/O Registers

- No changes

5.5 I/O Registers

- Changed title of Table 5-13
- Added ACPI I/O registers

5.6 Register Space Summary

- Updated with changes made to SYSCFG and PCICFG registers

Section 6.0 Electrical Ratings

- Changed section title

6.1 Absolute Maximum Ratings

- No changes

6.2 DC Characteristics

- Modified table, now have one DC table as opposed to four

6.3-6.7 AC Characteristics

- No changes made to the AC characteristic for the individual modules

6.8 AC Timing Diagrams

- No changes

Section 7.0 Test Mode Information

- New section to data book

Section 8.0 Mechanical Package Outline

- No changes

Table of Contents

Table of Contents (cont.)

Table of Contents (cont.)

Table of Contents (cont.)

Table of Contents (cont.)

Table of Contents (cont.)

Table of Contents (cont.)

Table of Contents (cont.)

Table of Contents (cont.)

List of Figures

List of Figures (cont.)

List of Tables

List of Tables (cont.)

List of Tables (cont.)

List of Tables (cont.)

List of Tables (cont.)

FireStar – 64-Bit CPU Single Chip Notebook Solution

1.0 Features

PCI Bus

- PCI supports sustained X-1-1-1... bursts, even to DRAM through an innovative mechanism. PCI operation can be concurrent with CPU/L2 cache and IDE operations.
- PCI clock generation eliminates the need for external PCI clock buffers in many designs and allows the PCI bus to be effectively power-managed.
- 3.3V or 5.0V PCI is supported on the FireStar PCI bus. If FireStar is configured for 3.3V operation, 5.0V-only PCI plug-in cards and docking stations can still be supported through a bridge device such as OPTi's 82C824 Cardbus Controller / Docking Solution, whose prefetch and post-write buffers off-load operations from the primary PCI bus. (Figure 1-1 illustrates the typical system architecture applicable when using the FireStar solution.)

DRAM Controller

- Provides BIOS with the means to automatically detect the DRAM type in use on each bank, whether fast page mode, EDO, or synchronous DRAM, allowing BIOS routines to efficiently program DRAM operation.

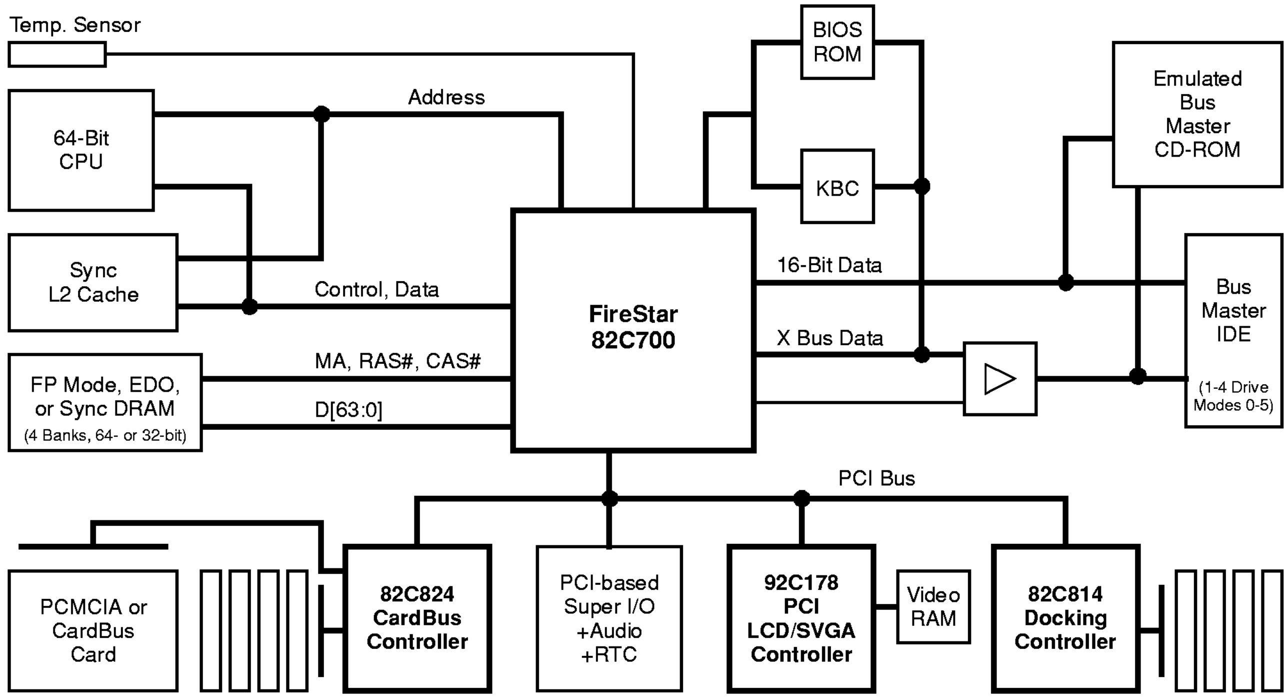
ISA Bus

- A full ISA bus is directly provided to support the keyboard controller, BIOS ROM, and Compact ISA peripheral devices for local ISA support with no TTL. When reduced ISA operation is selected, other FireStar pins become available for general purpose use.

Bus Mastering IDE

- FireStar supports two bus mastering IDE channels that function concurrently with operations on the CPU/L2 cache interface and PCI interface. Up to four drives are supported.
- An emulated bus mastering IDE feature allows IDE drives that are not commonly available as bus mastering devices, such as CD-ROM drives, to act as bus mastering drives. For example, a CD-ROM drive can transfer video data to DRAM while the CPU is decompressing the data and sending it to the graphics controller.

Figure 1-1 FireStar System Block Diagram



Thermal Management

- Fail-safe thermal management incorporates feedback logic that requires a very inexpensive external sensor circuit.
- Hardware monitors temperature directly and reliably, while the fail-safe aspect of the circuitry ensures that sensor component failure will automatically inhibit CPU clocking to prevent overheating.
- SMM code will be able to read (and display if desired) actual CPU temperature.

ACPI Implementation

- Microsoft Advanced Configuration and Power Interface (ACPI) is being implemented in the FireStar silicon. ACPI is a standard register interface for power management function jointly developed by Microsoft®, Intel®, and Toshiba®.

Miscellaneous

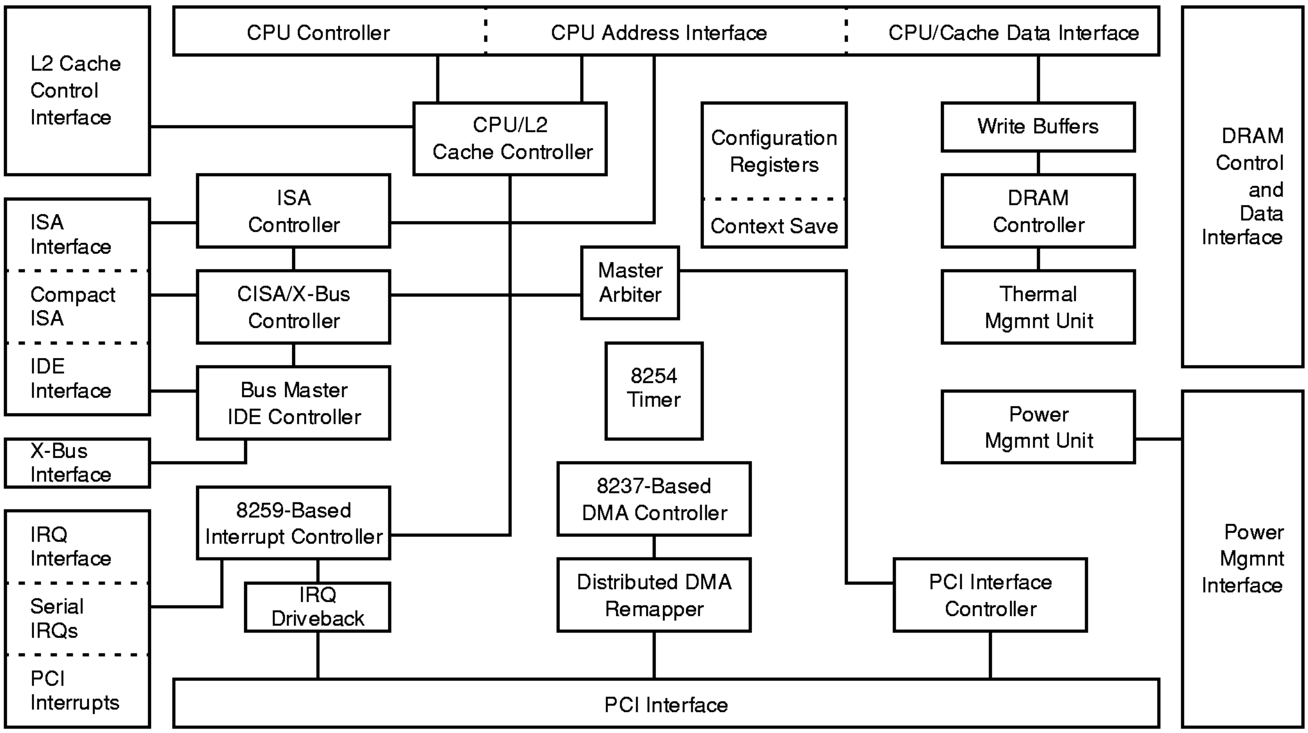
- The standard version of the chip can run at 3.3V, up to 66MHz on the CPU bus.
- A new Context Save Mode feature allows chip registers to be saved and restored more efficiently than ever before, requiring less SMM code and storage space.
- The OPTi Viper-N+ Power Management Unit is used, maintaining backward compatibility down to the register level with previously written support firmware.
- Serial IRQs are supported as an option for interrupts on PCI.
- Known devices in the system can be positively decoded on the PCI bus, eliminating the delay for subtractive decode and improving the efficiency of ISA operations.
- ISA bus cycle speed can be individually controlled to certain ISA device groups.
- Simple logic gate functions can be assigned to unused pins to eliminate the need for external TTL. Pin programming is far more flexible than ever possible on any other chip.

2.0 Overview

This data book describes the next generation Pentium® solution from the Mobile division of OPTi. The "FireStar" solution supports 64-bit 6x86-class CPUs and is very highly integrated, packaged in a single 432-pin BGA (Ball Grid Array). FireStar is intended as a low-cost yet high-performance notebook solution that can also be appropriate for use in certain desktop applications. Using FireStar in a full-featured, PCI-based notebook allows for designs with zero TTL.

Figure 2-1 shows the logic modules within the functional blocks of FireStar.

Figure 2-1 FireStar Logic Modules



3.0 Signal Definitions

3.1 Terminology/Nomenclature Conventions

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

The terms “assertion” and “negation” are used extensively. This is done to avoid confusion when working with a mixture of “active low” and “active high” signals. The term “assert”, or “assertion” indicates that a signal is active, independent of whether that level is represented by a high or low voltage. The term “negate”, or “negation” indicates that a signal is inactive.

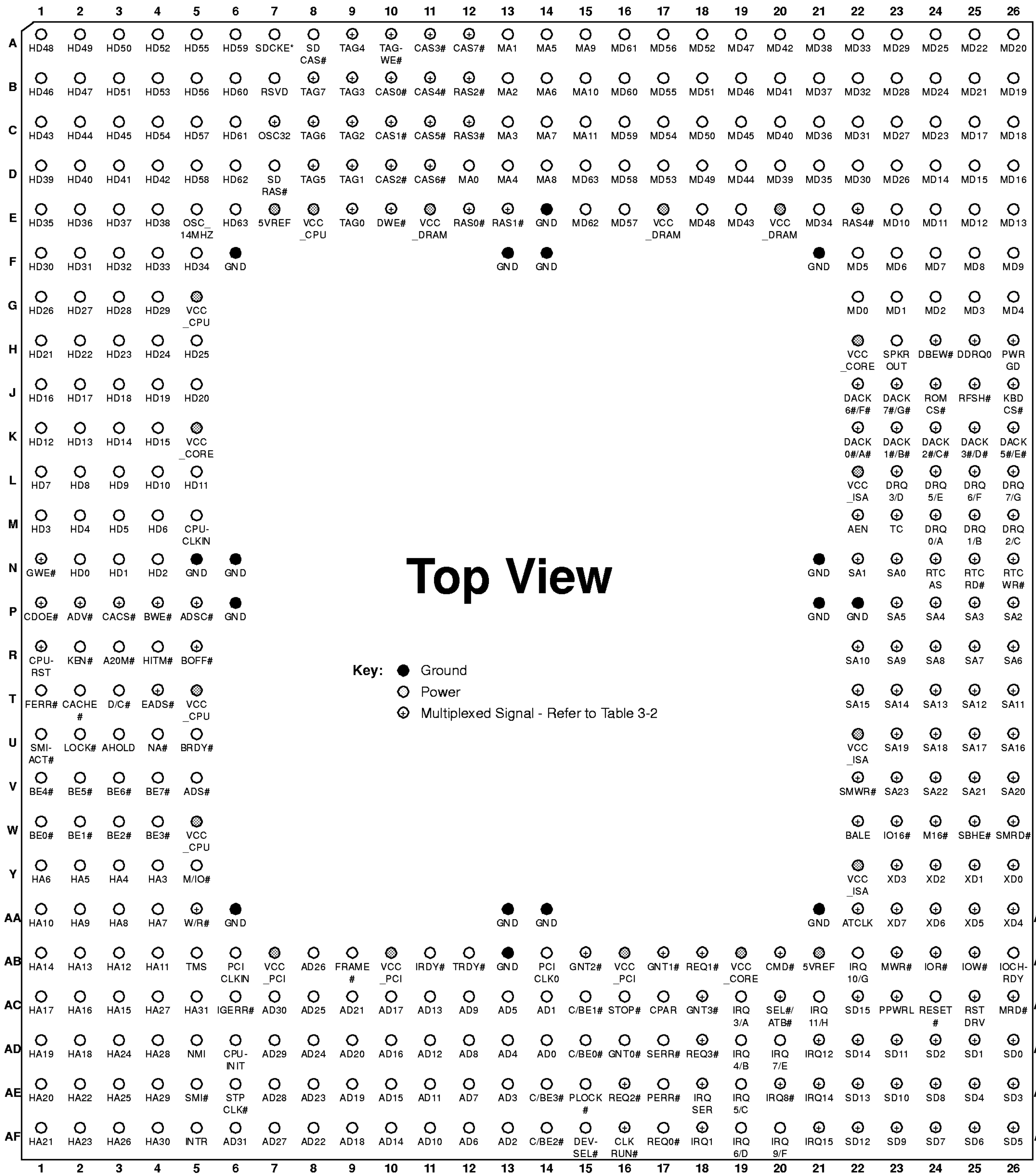
Some FireStar pins have more than one function. These pins can be time-multiplexed, have strap options, or can be selected via register programming. Included in the signal descriptions is a column titled “Selected By” which explains how to implement/invoke the various functions that a pin may have. The terms PCIDV0, PCIDV1, and SYSCFG relate to registers located in the PCI and System Configuration Register Spaces of FireStar. Refer to Section 5.0, "Register Descriptions" for more details regarding these register spaces and their access mechanisms.

The tables in this section use several common abbreviations. Table 3-1 lists the mnemonics and their meanings. Note that TTL/CMOS/Schmitt-trigger levels pertain to inputs only. Outputs are driven at CMOS levels.

Table 3-1 Signal Definitions Legend

Mnemonic	Description
CMOS	CMOS-level compatible
Dcdr	Decoder
Ext	External
G	Ground
I	Input
Int	Internal
I/O	Input/Output
Mux	Multiplexer
O	Output
OD	Open drain
P	Power
PD	Pull-down resistor
PU	Pull-up resistor
S	Schmitt-trigger
S/T/S	Sustain Tristate
TTL	TTL-level compatible

Figure 3-1 Pin Diagram



Note: *In FireStar ACPI pin A7 becomes SDCKE, where as in the non-ACPI version it is reserved. However, in both versions pin A7 is still used as part of the input address for the NAND tree test mode.

Table 3-2 Alphabetical Pin Cross-Reference List

Signal Name	Pin No.	Pin Type	Pwr Plane
A20M#	R3	O	CPU
AD0	AD14	I/O	PCI
AD1	AC14	I/O	PCI
AD2	AF13	I/O	PCI
AD3	AE13	I/O	PCI
AD4	AD13	I/O	PCI
AD5	AC13	I/O	PCI
AD6	AF12	I/O	PCI
AD7	AE12	I/O	PCI
AD8	AD12	I/O	PCI
AD9	AC12	I/O	PCI
AD10	AF11	I/O	PCI
AD11	AE11	I/O	PCI
AD12	AD11	I/O	PCI
AD13	AC11	I/O	PCI
AD14	AF10	I/O	PCI
AD15	AE10	I/O	PCI
AD16	AD10	I/O	PCI
AD17	AC10	I/O	PCI
AD18	AF9	I/O	PCI
AD19	AE9	I/O	PCI
AD20	AD9	I/O	PCI
AD21	AC9	I/O	PCI
AD22	AF8	I/O	PCI
AD23	AE8	I/O	PCI
AD24	AD8	I/O	PCI
AD25	AC8	I/O	PCI
AD26	AB8	I/O	PCI
AD27	AF7	I/O	PCI
AD28	AE7	I/O	PCI
AD29	AD7	I/O	PCI
AD30	AC7	I/O	PCI
AD31	AF6	I/O	PCI
ADS#	V5	I	CPU
ADSC#+PIO2	P5	O	CPU
ADV#+PIO3	P2	O	CPU
AEN+PPWR11	M22	I/O	ISA
AHOLD	U3	O	CPU
ATCLK+PCICLK4	AA22	O	ISA
BALE+PCICLK5	W22	O	ISA
BE0#	W1	I	CPU
BE1#	W2	I	CPU
BE2#	W3	I	CPU
BE3#	W4	I	CPU
BE4#	V1	I	CPU
BE5#	V2	I	CPU
BE6#	V3	I	CPU
BE7#	V4	I	CPU
BOFF#	R5	I/O	CPU
BRDY#	U5	O	CPU
BWE#	P4	O	CPU
CACHE#	T2	I	CPU

Signal Name	Pin No.	Pin Type	Pwr Plane
CACS#+DIRTY	P3	I/O	CPU
CAS0#+SDDQM0#	B10	O	DRAM
CAS1#+SDDQM1#	C10	O	DRAM
CAS2#+SDDQM2#	D10	O	DRAM
CAS3#+SDDQM3#	A11	O	DRAM
CAS4#+SDDQM4#	B11	O	DRAM
CAS5#+SDDQM5#	C11	O	DRAM
CAS6#+SDDQM6#	D11	O	DRAM
CAS7#+SDDQM7#	A12	O	DRAM
C/BE0#	AD15	I/O	PCI
C/BE1#	AC15	I/O	PCI
C/BE2#	AF14	I/O	PCI
C/BE3#	AE14	I/O	PCI
CDOE#+PIO0	P1	O	CPU
CLKRUN#+PIO6	AF16	I/O	PCI
CMD#+DIRTY+PCICLK3	AB20	I/O	ISA
CPAR	AC17	I/O	PCI
CPUCLKIN	M5	I	CPU
CPUINIT	AD6	O	CPU
CPURST+RSMRST	R1	O	CPU
D/C#	T3	I	CPU
DACK0#/DACKA#+PPWR4	K22	O	ISA
DACK1#/DACKB#+PPWR5	K23	O	ISA
DACK2#/DACKC#+PPWR6	K24	O	ISA
DACK3#/DACKD#+PPWR7	K25	O	ISA
DACK5#/DACKE#+PPWR13	K26	O	ISA
DACK6#/DACKF#+PPWR14	J22	O	ISA
DACK7#/DACKG#+PPWR15	J23	O	ISA
DBEW#+IDE1_DACK#+DWR#	H24	O	ISA
DDRQ0+PIO9	H25	I/O	ISA
DEVSEL#	AF15	I/O	PCI
DRQ0/DRQA+PIO25	M24	I/O	ISA
DRQ1/DRQB+PIO26	M25	I/O	ISA
DRQ2/DRQC+PIO27	M26	I/O	ISA
DRQ3/DRQD+PIO28	L23	I/O	ISA
DRQ5/DRQE+PIO29	L24	I/O	ISA
DRQ6/DRQF+PIO30	L25	I/O	ISA
DRQ7/DRQG+PIO31	L26	I/O	ISA
DWE#+SDWE#	E10	O	DRAM
EADS#+WB/WT#	T4	O	CPU
FERR#	T1	I	CPU
FRAME#	AB9	I/O	PCI
GND	AA6	G	
GND	AA13	G	
GND	AA14	G	
GND	AA21	G	

Signal Name	Pin No.	Pin Type	Pwr Plane
GND	AB13	G	
GND	E14	G	
GND	F6	G	
GND	F13	G	
GND	F14	G	
GND	F21	G	
GND	N5	G	
GND	N6	G	
GND	N21	G	
GND	P6	G	
GND	P21	G	
GND	P22	G	
GNT0#	AD16	O	PCI
GNT1#+PCICLK1	AB17	O	PCI
GNT2#+PCICLK2	AB15	O	PCI
GNT3#	AC18	O	PCI
GWE#+RAS5#	N1	O	CPU
HA3	Y4	I/O	CPU
HA4	Y3	I/O	CPU
HA5	Y2	I/O	CPU
HA6	Y1	I/O	CPU
HA7	AA4	I/O	CPU
HA8	AA3	I/O	CPU
HA9	AA2	I/O	CPU
HA10	AA1	I/O	CPU
HA11	AB4	I/O	CPU
HA12	AB3	I/O	CPU
HA13	AB2	I/O	CPU
HA14	AB1	I/O	CPU
HA15	AC3	I/O	CPU
HA16	AC2	I/O	CPU
HA17	AC1	I/O	CPU
HA18	AD2	I/O	CPU
HA19	AD1	I/O	CPU
HA20	AE1	I/O	CPU
HA21	AF1	I/O	CPU
HA22	AE2	I/O	CPU
HA23	AF2	I/O	CPU
HA24	AD3	I/O	CPU
HA25	AE3	I/O	CPU
HA26	AF3	I/O	CPU
HA27	AC4	I/O	CPU
HA28	AD4	I/O	CPU
HA29	AE4	I/O	CPU
HA30	AF4	I/O	CPU
HA31	AC5	I/O	CPU
HD0	N2	I/O	CPU
HD1	N3	I/O	CPU
HD2	N4	I/O	CPU
HD3	M1	I/O	CPU
HD4	M2	I/O	CPU
HD5	M3	I/O	CPU

Table 3-2 Alphabetical Pin Cross-Reference List (cont.)

Signal Name	Pin No.	Pin Type	Pwr Plane
HD6	M4	I/O	CPU
HD7	L1	I/O	CPU
HD8	L2	I/O	CPU
HD9	L3	I/O	CPU
HD10	L4	I/O	CPU
HD11	L5	I/O	CPU
HD12	K1	I/O	CPU
HD13	K2	I/O	CPU
HD14	K3	I/O	CPU
HD15	K4	I/O	CPU
HD16	J1	I/O	CPU
HD17	J2	I/O	CPU
HD18	J3	I/O	CPU
HD19	J4	I/O	CPU
HD20	J5	I/O	CPU
HD21	H1	I/O	CPU
HD22	H2	I/O	CPU
HD23	H3	I/O	CPU
HD24	H4	I/O	CPU
HD25	H5	I/O	CPU
HD26	G1	I/O	CPU
HD27	G2	I/O	CPU
HD28	G3	I/O	CPU
HD29	G4	I/O	CPU
HD30	F1	I/O	CPU
HD31	F2	I/O	CPU
HD32	F3	I/O	CPU
HD33	F4	I/O	CPU
HD34	F5	I/O	CPU
HD35	E1	I/O	CPU
HD36	E2	I/O	CPU
HD37	E3	I/O	CPU
HD38	E4	I/O	CPU
HD39	D1	I/O	CPU
HD40	D2	I/O	CPU
HD41	D3	I/O	CPU
HD42	D4	I/O	CPU
HD43	C1	I/O	CPU
HD44	C2	I/O	CPU
HD45	C3	I/O	CPU
HD46	B1	I/O	CPU
HD47	B2	I/O	CPU
HD48	A1	I/O	CPU
HD49	A2	I/O	CPU
HD50	A3	I/O	CPU
HD51	B3	I/O	CPU
HD52	A4	I/O	CPU
HD53	B4	I/O	CPU
HD54	C4	I/O	CPU
HD55	A5	I/O	CPU
HD56	B5	I/O	CPU
HD57	C5	I/O	CPU
HD58	D5	I/O	CPU
HD59	A6	I/O	CPU
HD60	B6	I/O	CPU
HD61	C6	I/O	CPU
HD62	D6	I/O	CPU
HD63	E6	I/O	CPU
HITM#	R4	I	CPU
IGERR#	AC6	I/O	CPU
INTR	AF5	O	CPU
IO16#+PIO18	W23	I/O	ISA
IOCHRDY	AB26	I/O	ISA
IOR#+IDE1_DRD#	AB24	I/O	ISA
IOW#+IDE1_DWR#	AB25	I/O	ISA
IRDY#	AB11	I/O	PCI
IRQ1+PIO10	AF18	I/O	PCI
IRQ3/IRQA	AC19	I	PCI
IRQ4/IRQB	AD19	I	PCI
IRQ5/IRQC	AE19	I	PCI
IRQ6/IRQD	AF19	I	PCI
IRQ7/IRQE	AD20	I	ISA
IRQ8#+PIO11	AE20	I/O	ISA
IRQ9/IRQF	AF20	I	ISA
IRQ10/IRQG	AB22	I	ISA
IRQ11/IRQH	AC21	I	ISA
IRQ12+PIO12	AD21	I/O	ISA
IRQ14+PIO13	AE21	I/O	ISA
IRQ15+SIN#	AF21	I	ISA
IRQSER+SDCKE+SOUT#	AE18	I/O	PCI
KBDCS#+PIO24+DRD#	J26	I/O	ISA
KEN#	R2	O	CPU
LOCK#	U2	I	CPU
M/IO#	Y5	I	CPU
M16#+PIO19	W24	I/O	ISA
MA0	D12	O	DRAM
MA1	A13	O	DRAM
MA2	B13	O	DRAM
MA3	C13	O	DRAM
MA4	D13	O	DRAM
MA5	A14	O	DRAM
MA6	B14	O	DRAM
MA7	C14	O	DRAM
MA8	D14	O	DRAM
MA9	A15	O	DRAM
MA10	B15	O	DRAM
MA11	C15	O	DRAM
MD0	G22	I/O	DRAM
MD1	G23	I/O	DRAM
MD2	G24	I/O	DRAM
MD3	G25	I/O	DRAM
MD4	G26	I/O	DRAM
MD5	F22	I/O	DRAM
MD6	F23	I/O	DRAM
MD7	F24	I/O	DRAM
MD8	F25	I/O	DRAM
MD9	F26	I/O	DRAM
MD10	E23	I/O	DRAM
MD11	E24	I/O	DRAM
MD12	E25	I/O	DRAM
MD13	E26	I/O	DRAM
MD14	D24	I/O	DRAM
MD15	D25	I/O	DRAM
MD16	D26	I/O	DRAM
MD17	C25	I/O	DRAM
MD18	C26	I/O	DRAM
MD19	B26	I/O	DRAM
MD20	A26	I/O	DRAM
MD21	B25	I/O	DRAM
MD22	A25	I/O	DRAM
MD23	C24	I/O	DRAM
MD24	B24	I/O	DRAM
MD25	A24	I/O	DRAM
MD26	D23	I/O	DRAM
MD27	C23	I/O	DRAM
MD28	B23	I/O	DRAM
MD29	A23	I/O	DRAM
MD30	D22	I/O	DRAM
MD31	C22	I/O	DRAM
MD32	B22	I/O	DRAM
MD33	A22	I/O	DRAM
MD34	E21	I/O	DRAM
MD35	D21	I/O	DRAM
MD36	C21	I/O	DRAM
MD37	B21	I/O	DRAM
MD38	A21	I/O	DRAM
MD39	D20	I/O	DRAM
MD40	C20	I/O	DRAM
MD41	B20	I/O	DRAM
MD42	A20	I/O	DRAM
MD43	E19	I/O	DRAM
MD44	D19	I/O	DRAM
MD45	C19	I/O	DRAM
MD46	B19	I/O	DRAM
MD47	A19	I/O	DRAM
MD48	E18	I/O	DRAM
MD49	D18	I/O	DRAM
MD50	C18	I/O	DRAM
MD51	B18	I/O	DRAM
MD52	A18	I/O	DRAM
MD53	D17	I/O	DRAM
MD54	C17	I/O	DRAM
MD55	B17	I/O	DRAM
MD56	A17	I/O	DRAM
MD57	E16	I/O	DRAM
MD58	D16	I/O	DRAM

Table 3-2 Alphabetical Pin Cross-Reference List (cont.)

Signal Name	Pin No.	Pin Type	Pwr Plane
MD59	C16	I/O	DRAM
MD60	B16	I/O	DRAM
MD61	A16	I/O	DRAM
MD62	E15	I/O	DRAM
MD63	D15	I/O	DRAM
MRD#+IDE1_DCS3#	AC26	I/O	ISA
MWR#+IDE1_DCS1#	AB23	I/O	ISA
NA#	U4	O	CPU
NMI	AD5	O	CPU
OSC_14MHZ	E5	I	CPU
OSC32	C7	I	CPU
PCICLK0	AB14	O	PCI
PCICLKIN	AB6	I	CPU
PERR#	AE17	I/O	PCI
PLOCK#	AE15	I/O	PCI
PPWRL+PPWR0#	AC23	I/O	ISA
PWRGD	H26	I	ISA
RAS0#+SDCS0#	E12	O	DRAM
RAS1#+SDCS1#+PIO5	E13	O	DRAM
RAS2#+SDCS2#+PIO4	B12	O	DRAM
RAS3#+SDCS3#+MA12	C12	O	DRAM
RAS4#+MA12	E22	O	DRAM
RFSH#+PPWR12	J25	I/O	ISA
REQ0#	AF17	I	PCI
REQ1#+PIO7	AB18	I/O	PCI
REQ2#+PIO8	AE16	I/O	PCI
REQ3#	AD18	I	PCI
RESET#	AC24	O	ISA
ROMCS#+PIO23+ ROMCS#/KBDCS#	J24	I/O	ISA
RSTDRV+PIO15	AC25	I/O	ISA
RTCAS+IDE1_DA0	N24	I/O	ISA
RTCRD#+IDE1_DA1	N25	I/O	ISA
RTCWR#+IDE1_DA2	N26	I/O	ISA
SA0+IDE1_DD0	N23	I/O	ISA
SA1+IDE1_DD1	N22	I/O	ISA
SA2+IDE1_DD2	P26	I/O	ISA
SA3+IDE1_DD3	P25	I/O	ISA
SA4+IDE1_DD4	P24	I/O	ISA
SA5+IDE1_DD5	P23	I/O	ISA
SA6+IDE1_DD6	R26	I/O	ISA
SA7+IDE1_DD7	R25	I/O	ISA
SA8+IDE1_DD8	R24	I/O	ISA
SA9+IDE1_DD9	R23	I/O	ISA
SA10+IDE1_DD10	R22	I/O	ISA
SA11+IDE1_DD11	T26	I/O	ISA
SA12+IDE1_DD12	T25	I/O	ISA
SA13+IDE1_DD13	T24	I/O	ISA
SA14+IDE1_DD14	T23	I/O	ISA
SA15+IDE1_DD15	T22	I/O	ISA
SA16+PIO16	U26	I/O	ISA
SA17+PIO17	U25	I/O	ISA

Signal Name	Pin No.	Pin Type	Pwr Plane
SA18+PPWR8	U24	I/O	ISA
SA19+PPWR9	U23	I/O	ISA
SA20+PPWR0	V26	I/O	ISA
SA21+PPWR1	V25	I/O	ISA
SA22+PPWR2	V24	I/O	ISA
SA23+PPWR3	V23	I/O	ISA
SBHE#+PIO20	W25	I/O	ISA
SD0+MAD0	AD26	I/O	ISA
SD1+MAD1	AD25	I/O	ISA
SD2+MAD2	AD24	I/O	ISA
SD3+MAD3	AE26	I/O	ISA
SD4+MAD4	AE25	I/O	ISA
SD5+MAD5	AF26	I/O	ISA
SD6+MAD6	AF25	I/O	ISA
SD7+MAD7	AF24	I/O	ISA
SD8+MAD8	AE24	I/O	ISA
SD9+MAD9	AF23	I/O	ISA
SD10+MAD10	AE23	I/O	ISA
SD11+MAD11	AD23	I/O	ISA
SD12+MAD12	AF22	I/O	ISA
SD13+MAD13	AE22	I/O	ISA
SD14+MAD14	AD22	I/O	ISA
SD15+MAD15	AC22	I/O	ISA
SDCAS#	A8	O	DRAM
SDRAS#	D7	O	DRAM
SEL#/ATB#+SDCKE+ PIO14	AC20	I/O	ISA
SERR#	AD17	I/O	PCI
SMI#	AE5	O	CPU
SMIACT#	U1	I	CPU
SMRD#+PIO21	W26	I/O	ISA
SMWR#+PIO22	V22	I/O	ISA
SPKOUT	H23	I/O	ISA
STOP#	AC16	I/O	PCI
STPCLK#	AE6	O	CPU
TAG0+CAS0#	E9	I/O	DRAM
TAG1+CAS1#+START#	D9	I/O	DRAM
TAG2+CAS2#+START#	C9	I/O	DRAM
TAG3+CAS3#+SBOFF#	B9	I/O	DRAM
TAG4+CAS4#+SDCKE	A9	I/O	DRAM
TAG5+CAS5#+DWE#	D8	I/O	DRAM
TAG6+CAS6#+SDCAS#	C8	I/O	DRAM
TAG7+CAS7#+SDRAS#	B8	I/O	DRAM
TAGWE#+PIO1	A10	O	DRAM
TC+PPWR10	M23	I/O	ISA
RSVD	B7	I	CPU
SDCKE	A7	O	CPU
TMS	AB5		CPU
TRDY#	AB12	I/O	PCI
VCC_CORE	H22	P	
VCC_CORE	K5	P	
VCC_CORE	AB19	P	

Signal Name	Pin No.	Pin Type	Pwr Plane
VCC_CPU	E8	P	
VCC_CPU	G5	P	
VCC_CPU	T5	P	
VCC_CPU	W5	P	
VCC_DRAM	E11	P	
VCC_DRAM	E17	P	
VCC_DRAM	E20	P	
VCC_ISA	L22	P	
VCC_ISA	U22	P	
VCC_ISA	Y22	P	
VCC_PCI	AB7	P	
VCC_PCI	AB10	P	
VCC_PCI	AB16	P	
W/R#+INV	AA5	I/O	CPU
XD0+IDE_DWR#	Y26	I/O	ISA
XD1+IDE_DRD#	Y25	I/O	ISA
XD2+IDE_DA0	Y24	I/O	ISA
XD3+IDE_DA1	Y23	I/O	ISA
XD4+IDE_DA2	AA26	I/O	ISA
XD5+IDE_DDACK#	AA25	I/O	ISA
XD6+IDE_DCS1#	AA24	I/O	ISA
XD7+IDE_DCS3#	AA23	I/O	ISA
5VREF	AB21	P	
5VREF	E7	P	

Power Plane Key:
CORE = 3.3V Only
CPU = 3.3V (and 2.5V in future revisions)
DRAM = 3.3V or 5.0V
ISA = 3.3V or 5.0V
PCI = 3.3V or 5.0V

Note: The pins listed below are 5.0V tolerant inputs, even when their power plane is connected to 3.3V as long as the 5VREF pins of FireStar are connected to +5.0V:

- OSC32
- OSC_14MHZ
- CACS#+DIRTY
- PCICLK
- IRQA
- IRQB
- IRQC
- IRQD
- IRQ1

3.2 Signal Descriptions

3.2.1 CPU Interface Signals Set

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
Host Data Bus				
HD[63:0]	Refer to Table 3-2	I/O (4mA)		**Host Data Bus Lines 63 through 0:** Provides a 64-bit data path to the CPU.
CPU Address				
HA[31:3]	Refer to Table 3-2	I/O (4mA)		**Host Address Bus Lines 31 through 3:** HA[31:3] are the address lines of the CPU bus. HA[31:3] are connected to CPU lines A[31:3]. Along with the byte enable signals, HA[31:3] define the physical area of memory or I/O being accessed. During CPU cycles, the HA[31:3] lines are inputs. They are used for address decoding and second level cache tag lookup sequences. During inquire cycles, the HA[31:5] lines are outputs to the CPU to snoop the first level cache tags. They also are outputs to the L2 cache.
BE[7:0]#	V4:V1, W4:W1	I		**Byte Enables 7 through 0:** Selects the active byte lanes on HD[63:0].
NMI	AD5 Strap option pin, refer to Table 3-7	O (4mA)		**Non-Maskable Interrupt:** This signal is activated when a parity error from a local memory read is detected or when the IOCHK# signal from the ISA bus is asserted and the corresponding control bit in Port B is also enabled.
INTR	AF5 Strap option pin, refer to Table 3-7	O (4mA)		**Interrupt Request:** INTR is driven to signal the CPU that an interrupt request is pending and needs to be serviced. The interrupt controller must be programmed following a reset to ensure that INTR is at a known state.
FERR#	T1	I		**Floating Point Coprocessor Error:** This input causes two operations to occur. IRQ13 is triggered and IGERR# is enabled. An I/O write to Port F0h will set IGERR# low when FERR# is low.
IGERR#	AC6 Strap option pin, refer to Table 3-7	I/O (4mA)		**Ignore Coprocessor Error:** Normally high, IGERR# will go low after FERR# goes low and an I/O write to Port 0F0h occurs. When FERR# goes high, IGERR# is driven high.
CPU Control/Status				
CPUINIT	AD6	O		**CPU Initialize:** A shutdown cycle or a low-to-high transition of I/O Port 092h bit 0 will trigger CPUINIT. If keyboard emulation is enabled (default), a CPUINIT will be generated when a Port 064h write cycle with data FEh is decoded. If keyboard emulation has been disabled, then this signal will be triggered when it sees the KBRST from the keyboard.

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
M/IO#	Y5	I		**Memory/Input-Output:** M/IO#, D/C#, and W/R# define CPU bus cycles. Interrupt acknowledge cycles are forwarded to the PCI bus as PCI interrupt acknowledge cycles. All I/O cycles and any memory cycles that are not directed to memory controlled by the DRAM interface are forwarded to PCI.
D/C#	T3	I		**Data/Control**: D/C#, M/IO#, and W/R# define CPU bus cycles. (See M/IO# definition above.)
W/R#	AA5	I/O (4mA)	Cycle Multiplexed	**Write/Read:** W/R#, D/C#, and M/IO# define CPU bus cycles. (See M/IO# definition above.)
INV		O (4mA)		**Invalidate:** Pin AA5 also serves as an output signal and is used as INV for L1 cache during an inquire cycle.
ADS#	V5	I		**Address Strobe:** The CPU asserts ADS# to indicate that a new bus cycle is beginning. ADS# is driven active in the same clock as the address, byte enables, and cycle definition signals. ADS# has an internal pull-up resistor that is disabled when the system is in the Suspend mode.
BRDY#	U5	O (4mA)		**Burst Ready:** BRDY# indicates that the system has responded in one of three ways: 1) Valid data has been placed on the CPU data bus in response to a read, 2) CPU write data has been accepted by the system, or 3) the system has responded to a special cycle.
NA#	U4	O (4mA)		**Next Address:** This signal is connected to the CPU's NA# pin to request pipelined addressing for local memory cycle. FireStar asserts NA# for one clock when the system is ready to accept a new address from the CPU, even if all data transfers for the current cycle have not completed.
KEN#	R2	O (4mA)		**Cache Enable:** This pin is connected to the KEN# input of the CPU and is used to determine whether the current cycle is cacheable.
EADS#	T4	O (4mA)	Cycle Multiplexed	**External Address Strobe:** This output indicates that a valid address has been driven onto the CPU address bus by an external device. This address will be used to perform an internal cache inquiry cycle when the CPU samples EADS# active.
WB/WT#				**Writeback/Write-Through**: Pin T4 is also used to control writeback or write-through policy for the primary cache during CPU cycles.
HITM#	R4	I		**Hit Modified:** Indicates that the CPU has had a hit on a modified line in its internal cache during an inquire cycle. It is used to prepare for writeback.

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
CACHE#	T2	I		**Cacheability:** This input is connected to the CACHE# pin of the CPU. It goes active during a CPU initiated cycle to indicate when, an internal cacheable read cycle or a burst writeback cycle, occurs.
AHOLD	U3	O (4mA)		**Address Hold**: This signal is used to tristate the CPU address bus for internal cache snooping.
LOCK#	U2	I		**CPU Bus Lock:** The processor asserts LOCK# to indicate the current bus cycle is locked. It is used to generate PLOCK# for the PCI bus. LOCK# has an internal pull-down resistor that is engaged when HLDA is active.
BOFF#	R5 Strap option pin, refer to Table 3-7	O (4mA)		**Back-off:** This pin is connected to the BOFF# input of the CPU.
CPURST	R1	O (4mA)	(Always)	**CPU Reset:** This signal generates a hard reset to the CPU whenever the PWRGD input goes active.
RSMRST			SYSCFG ADh[5] = 1	**Resume Reset:** Generates a hard reset to the CPU on resuming from Suspend mode.
Host Power Control				
SMI#	AE5	O (4mA)		**System Management Interrupt:** This signal is used to request System Management Mode (SMM) operation.
SMIACT#	U1	I		**System Management Interrupt Active:** The CPU asserts SMIACT# in response to the SMI# signal to indicate that it is operating in System Management Mode (SMM).
STPCLK#	AE6	O (4mA)		**Stop Clock:** This signal is connected to the STPCLK# input of the CPU. It causes the CPU to get into the STPGNT# state.
L2 Cache Control				
CDOE#	P1	O (4mA)	PCIDV1 80h = 00h	**Cache Output Enable:** This signal is connected to the output enables of the SRAMs of the L2 cache in both banks to enable data read.
PIO0			PCIDV1 80h $\neq$ 00h	**Programmable Input/Output 0:** Due to the critical timing required for the functionality of this pin, it can be programmed only as an output. See Section 3.3, "Programmable I/O Pins", on page 33 for more details.

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
CACS#	P3	O (4mA)	See SYSCFG 16h[7,5] bit descriptions on page 266	**Cache Chip Select:** This pin is connected to the chip selects of the SRAMs in the L2 cache to enable data read/write operations. If not used, the CS# lines of the cache should be tied low.
DIRTY		I/O (4mA)		**Tag Dirty Bit:** This separate dirty bit allows the tag data to be 8 bits wide instead of 7. DIRTY is a 5.0V tolerant input, even when its power plane is connected to 3.3V as long as the 5VREF pins of FireStar are connected to +5.0V.
BWE#	P4	O (4mA)		**Byte Write Enable:** Write command to L2 cache indicating that only bytes selected by BE[7:0]# will be written.
GWE#	N1	O (4mA)	SYSCFG 19h[7] = 0	**Global Write Enable:** Write command to L2 cache indicating that all bytes will be written.
RAS5#			SYSCFG 19h[7] = 1	**Row Address Strobe Bit 5:** Each RAS# signal corresponds to a unique DRAM bank. Depending on the kind of DRAM modules being used, this signal may or may not need to be buffered externally. This signal, however, should be connected to the corresponding DRAM RAS# line through a damping resistor.
TAG0	E9	I/O (4mA)	SYSCFG 11h[3] = 0	**Tag RAM Data Bit 0:** This input signal becomes an output whenever TAGWE# is activated to write a new tag to the Tag RAM.
CAS0#		O (4mA)	SYSCFG 11h[3] = 1	**Column Address Strobe Bit 0 (2nd copy)**
TAG1	D9	I/O (4mA)	SYSCFG 00h[5] = 0 11h[3] = 0	**Tag RAM Data Bit 1:** This input signal becomes an output whenever TAGWE# is activated to write a new tag to the Tag RAM.
CAS1#		O (4mA)	SYSCFG 11h[3] = 1	**Column Address Strobe Bit 1 (2nd copy)**
START#		O (4mA)	SYSCFG 00h[5] = 1	**Start:** If using the Sony cache module, then this pin is connected to the START# output from the Sony SONIC2-WP module. If using the Sony cache module, then TAG1 and TAG2 are connected to the START# output from the module and TAG3 is connected to the BOFF# output from the module. The remaining TAG bits are unused.

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
TAG2	C9	I/O (4mA)	SYSCFG 00h[5] = 0 11h[3] = 0	**Tag RAM Data Bit 2:** This input signal becomes an output whenever TAGWE# is activated to write a new tag to the Tag RAM.
CAS2#		O (4mA)	SYSCFG 11h[3] = 1	**Column Address Strobe Bit 2 (2nd copy)**
START#		O (4mA)	SYSCFG 00h[5] = 1	**Start:** If using the Sony cache module, then this pin is connected to the START# output from the Sony SONIC2-WP module. If using the Sony cache module, then TAG1 and TAG2 are connected to the START# output from the module and TAG3 is connected to the BOFF# output from the module. The remaining TAG bits are unused
TAG3	B9	I/O (4mA)	SYSCFG 00h[5] = 0 11h[3] = 0	**Tag RAM Data Bit 3:** This input signal becomes an output whenever TAGWE# is activated to write a new tag to the Tag RAM.
CAS3#		O (4mA)	SYSCFG 11h[3] = 1	**Column Address Strobe Bit 3 (2nd copy)**
SBOFF#		O (4mA)	SYSCFG 00h[5] = 1	**Sony Back-off:** For use with Sony SONIC-2WP cache module.
TAG4	A9	I/O (4mA)	SYSCFG 11h[3] = 0	**Tag RAM Data Bit 4:** This input signal becomes an output whenever TAGWE# is activated to write a new tag to the Tag RAM.
CAS4#		O (4mA)	SYSCFG 11h[3] = 1	**Column Address Strobe Bit 4 (2nd copy)**
SDCKE			PCIDV1 53h[7] = 1	**SDRAM Clock Enable:** This signal is asserted to put the SDRAM into a “Stop” state. The BIOS can program FireStar to assert this signal only in Suspend mode.
TAG5	D8	I/O (4mA)	SYSCFG 11h[3] = 0	**Tag RAM Data Bit 5:** This input signal becomes an output whenever TAGWE# is activated to write a new tag to the Tag RAM.
CAS5#		O (4mA)	SYSCFG 11h[3] = 1	**Column Address Strobe Bit 5 (2nd copy)**
DWE#		O (4mA)	PCIDV1 53h[7] = 1	**DRAM Write Enable (2nd copy)**
TAG6	C8	I/O (4mA)	SYSCFG 11h[3] = 0	**Tag RAM Data Bit 6:** This input signal becomes an output whenever TAGWE# is activated to write a new tag to the Tag RAM.
CAS6#		O (4mA)	SYSCFG 11h[3] = 1	**Column Address Strobe Bit 6 (2nd copy)**
SDCAS#		O (4mA)	PCIDV1 53h[7] = 1	**SDRAM Column Address Strobe (2nd copy)**

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
TAG7	B8	I/O (4mA)	SYSCFG 11h[3] = 0	**Tag RAM Data Bit 7:** This input signal becomes an output whenever TAGWE# is activated to write a new tag to the Tag RAM.
CAS7#		O (4mA)	SYSCFG 11h[3] = 1	**Column Address Strobe Bit 7 (2nd copy)**
SDRAS#		O (4mA)	PCIDV1 53h[7] = 1	**SDRAM Row Address Strobe (2nd copy)**
TAGWE#	A10	O (4mA)	PCIDV1 81h = 00h	**Tag RAM Write Enable:** This control strobe is used to update the Tag RAM with the valid tag of the new cache line that replaces the current one during external cache read miss cycles.
PIO1			PCIDV1 81h ≠ 00h	**Programmable Input/Output 1:** Due to the critical timing required for the functionality of this pin, it can be programmed only as an output. See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
ADSC#	P5	O (4mA)	PCIDV1 82h = 00h	**Controller Address Strobe:** For a synchronous L2 cache operation, this pin is connected to the ADSC# input of the synchronous SRAMs.
PIO2			PCIDV1 82h ≠ 00h	**Programmable Input/Output 2:** Due to the critical timing required for the functionality of this pin, it can be programmed only as an output. See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
ADV#	P2	O (4mA)	PCIDV1 83h = 00h	**Advance Output:** For synchronous cache L2 operation, this pin becomes the advance output and is connected to the ADV# input of the synchronous SRAMs.
PIO3			PCIDV1 83h ≠ 00h	**Programmable Input/Output 3:** Due to the critical timing required for the functionality of this pin, it can be programmed only as an output. See Section 3.3, "Programmable I/O Pins", on page 33 for more details.

3.2.2 DRAM and PCI Interface Signals Set

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
DRAM Interface				
RAS0#	E12	O (8/12mA)	Cycle Multiplexed	**Row Address Strobe 0:** Each RAS# signal corresponds to a unique DRAM bank. Depending on the kind of DRAM modules being used, this signal may or may not need to be buffered externally. This signal, however, should be connected to the corresponding DRAM RAS# line through a damping resistor.
SDCS0#				**SDRAM Chip Select Line 0:** Each SDCS# output corresponds to a unique SDRAM Bank. When active, the SDRAM will accept the command from FireStar. These outputs must be connected to the SDRAM banks through a damping resistor.
RAS1#	E13	O (8/12mA)	Cycle Multiplexed if PCIDV1 85h = 00h	**Row Address Strobe 1:** Refer to RAS0# signal description.
SDCS1#				**SDRAM Chip Select Line 1:** Refer to SDCS0# description.
PIO5			PCIDV1 85h ≠ 00h	**Programmable Input/Output 5:** Due to the critical timing required for the functionality of this pin, it can be programmed only as an output. See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
RAS2#	B12	O (8/12mA)	Cycle Multiplexed if PCIDV1 84h = 00h	**Row Address Strobe 2:** Refer to RAS0# signal description.
SDCS2#				**SDRAM Chip Select Line 2:** Refer to SDCS0# description.
PIO4			PCIDV1 84h ≠ 00h	**Programmable Input/Output 4:** Due to the critical timing required for the functionality of this pin, it can be programmed only as an output. See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
RAS3#	C12	O (8/12mA)	Cycle Multiplexed	**Row Address Strobe 3:** Refer to RAS0# signal description.
SDCS3#				**SDRAM Chip Select Line 3:** Refer to SDCS0# description.
MA12			PCIDV1 53h[6:5] = 01	**Memory Address Bus Line 12**
RAS4#	E22	O (8/12mA)	SYSCFG 19h[3] = 1	**Row Address Strobe 4 (primary copy):** Refer to RAS0# signal description.
MA12			SYSCFG 19h[3] = 1 PCIDV1 53h[6:5] = 10	**Memory Address Bus Line 12**

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
CAS[7:0]#	A12, D11, C11, B11, A11, D10, C10, B10	O (8mA)	Cycle Multiplexed	**Column Address Strobe Lines 7 through 0 (primary copies):** The CAS[7:0]# outputs correspond to the eight bytes for each DRAM bank. Each DRAM bank has a 64-bit data bus. These signals are typically connected directly to the DRAM's CAS# inputs through a damping resistor.
SDDQM[7:0]#				**SDRAM Data Mask Control Bits 7 through 0:** During SDRAM read cycles, these outputs control whether the DRAM output buffers are driven on the MD bus or not. During SDRAM write cycles, these outputs control whether or not MD data will be written into the memory device.
SDCAS#	A8	O		**SDRAM Column Address Strobe (primary copy):** This output is part of the SDRAM command combination. This pin should be connected to the SDRAM through a damping resistor.
SDRAS#	D7	O		**SDRAM Row Address Strobe (primary copy):** This output is part of the SDRAM command combination. This pin should be connected to the SDRAM through a damping resistor.
DWE#	E10	O (8mA)	Cycle Multiplexed	**DRAM Write Enable (primary copy):** This signal is the common write enable for all 64 bits of DRAM if either fast page mode or EDO DRAMs are used. This signal can be buffered externally before connection to the WE# input of the DRAMs.
SDWE#				**SDRAM Write Enable:** This output is the write enable signal for SDRAM.
MA[11:0]	Refer to Table 3-2	O (8/12mA)		**Memory Address Bus Lines 11 through 0:** Multiplexed row/column address lines to the DRAMs. Depending on the kind of DRAM modules being used, these signals may or may not need to be buffered externally. MA12 is optionally available instead of RAS3# or RAS4#.
MD[63:32]	Refer to Table 3-2	I/O (4mA)		**Higher Order Memory Data Bus:** These pins are connected directly to the higher order DRAM data bus.
MD[31:0]	Refer to Table 3-2	I/O (4mA)		**Lower Order Memory Data Bus:** These pins are connected directly to the lower order DRAM data bus.
PCI Bus Interface				
AD[31:0]	Refer to Table 3-2	I/O (PCI)		**PCI Address and Data:** AD[31:0] are bidirectional address and data lines for the PCI bus. The AD[31:0] signals sample or drive the address and data on the PCI bus.

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
C/BE[3:0]#	AE14, AF14, AC15, AD15	I/O (PCI)		**PCI Bus Command and Byte Enables:** During the address phase of a transaction, C/BE[3:0]# define the PCI command. During the data phase, C/BE[3:0]# are used as the PCI byte enables. The PCI commands indicate the current cycle type, and the PCI byte enables indicate which byte lanes carry meaningful data. FireStar drives C/BE[3:0]# as an initiator of a PCI bus cycle and monitors C/BE[3:0]# as a target.
CPAR	AC17	I/O (PCI)		**Calculated Parity Signal:** PAR is “even” parity and is calculated on 36 bits - AD[31:0] plus C/BE[3:0]#. PAR is generated for address and data phases and is only guaranteed to be valid on the PCI clock after the corresponding address or data phase.
FRAME#	AB9	I/O (PCI)		**Cycle Frame:** FRAME# is driven by the current bus master to indicate the beginning and duration of an access. FRAME# is asserted to indicate that a bus transaction is beginning. FRAME# is an input when FireStar is the target and an output when it is the initiator.
IRDY#	AB11	I/O (PCI)		**Initiator Ready:** IRDY# indicates FireStar’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 each clock that both IRDY# and TRDY# are sampled asserted. IRDY# is an input to when FireStar is the target and an output when it is the initiator.
TRDY#	AB12	I/O (PCI)		**Target Ready:** TRDY# indicates FireStar’s ability to complete the current data phase of the transaction. It is used in conjunction with IRDY#. A data phase is completed on each clock that TRDY# and IRDY# are both sampled asserted. TRDY# is an input when FireStar is the initiator and an output when it is the target.
DEVSEL#	AF15	I/O (PCI)		**Device Select:** FireStar asserts DEVSEL# to claim a PCI transaction. As an output, FireStar asserts DEVSEL# when it samples configuration cycles to the configuration registers. FireStar also asserts DEVSEL# when an internal IPC address is decoded. As an input, DEVSEL# indicates the response to a transaction. If no slave claims the cycle, FireStar will assert DEVSEL# to terminate the cycle.
STOP#	AC16	I/O (PCI)		**Stop:** STOP# indicates that FireStar, as a target, is requesting a master to stop the current transaction. As a master, STOP# causes FireStar to stop the current transaction. STOP# is an output when FireStar is a target and an input when it is the initiator.

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
PLOCK#	AE15	I/O (PCI)		**PCI Lock:** PLOCK# is used to indicate an atomic operation that may require multiple transactions to complete. When PLOCK# is asserted, non-exclusive transactions may proceed to an address that is not currently locked. Control of PLOCK# is obtained under its own protocol in conjunction with PGNT#.
SERR#	AD17	I/O (PCI)		**System Error:** SERR# can be pulsed active by any PCI device that detects a system error condition. Upon sampling SERR# active, FireStar generates a non-maskable interrupt (NMI) to the 3.3V Pentium CPU.
PERR#	AE17	I/O (4mA)		**Parity Error:** PERR# may be pulsed by any agent that detects a parity error during an address phase, or by the master, or by the selected target during any data phase in which the AD[31:0] lines are inputs. Upon sampling PERR# active, FireStar generates a non-maskable interrupt (NMI) to the 3.3V Pentium CPU.
PCICLKIN	AB6	I		**PCI Clock Input:** Master PCI clock input on the CPU power plane. PCICLKIN is a 5.0V tolerant input, even when its power plane is connected to 3.3.V as long as the 5VREF pins of FireStar are connected to +5.0V.
CLKRUN#	AF16	I/O (PCI)	PCIDV1 86h = 00h	**Clock Run:** CLKRUN# is an I/O sustained tristate signal and follows the PCI 2.1 defined protocol. When a PCI device pulls CLKRUN# low, FireStar enables PCICLK by asserting *CLKOE (PIO option)* high. FireStar maintains control of CLKRUN# and will keep it low as long as it intends to keep the clock running. FireStar will attempt to turn off the PCI clock to PCI devices whenever software enables APM Doze mode (setting *SYSCFG 50h[3] = 1*). Note that the FireStar PCICLK input must not be turned off. A weak external pull-up is required. Also refer to the *CLKOE* signal description in Section 3.3, Programmable I/O Pins.
PIO6			PCIDV1 86h ≠ 00h	**Programmable Input/Output 6:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
REQ0#	AF17	I		**PCI Bus Request 0:** REQ# is used by PCI bus masters to request control of the bus.
GNT0#	AD16	O (PCI)		**PCI Bus Grant 0:** GNT# is returned to PCI bus masters asserting REQ#, when the bus becomes available.

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
REQ1#	AB18	I	PCIDV1 87h = 00h	**PCI Bus Request 1:** REQ# is used by PCI bus masters to request control of the bus.
PIO7		I/O (4mA)	PCIDV1 87h ≠ 00h	**Programmable Input/Output 7:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
PCICLK0	AB14 Strap option pin, refer to Table 3-7	O (PCI)		**PCI Clock Output 0:** This PCI clock output is always available.
GNT1#	AB17	O (PCI)	Default	**PCI Bus Grant 1:** GNT# is returned to PCI bus masters asserting REQ#, when the bus becomes available.
PCICLK1		O (4mA)	RTCRD# strap option	**PCI Clock Output 1**
REQ2#	AE16	I	PCIDV1 88h = 00h	**PCI Bus Request 2:** REQ# is used by PCI bus masters to request control of the bus.
PIO8		I/O (4mA)	PCIDV1 88h ≠ 00h	**Programmable Input/Output 8:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
GNT2#	AB15	O (PCI)	Default	**PCI Bus Grant 2:** GNT# is returned to PCI bus masters asserting REQ#, when the bus becomes available.
PCICLK2			RTCWR# strap option	**PCI Clock Output 2**
REQ3#	AD18	I		**PCI Bus Request 3:** REQ# is used by PCI bus masters to request control of the bus.
GNT3#	AC18	O (PCI)		**PCI Bus Grant 3:** GNT# is returned to PCI bus masters asserting REQ#, when the bus becomes available.

3.2.3 IDE Interface Signal Set

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
Bus Master IDE Interface				
DBEW#	H24 Strap option pin, refer to Table 3-7	O (4mA)	Default	**Drive W Buffer Control**
IDE1_DACK#			RTCAS:A20M# strap option	**DDACK# for Second IDE Cable**
DWR#			PCIDV1 4Fh[1] = 1	**Drive Write Signal**

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
DDRQ0	H25	I/O (4mA)	PCIDV1 89h = 00h	**Drive Cable A DMA Request**
PIO9			PCIDV1 89h ≠ 00h	**Programmable Input/Output 9:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
Clock and Reset Interface				
RESET#	AC24	O (8mA)		**System Reset:** When asserted, this signal resets the CPU. RESET# is asserted in response to a PWRGD only and is guaranteed to be active for 1ms such that CLK and VCC are stable. If RSTDRV is programmed to toggle in Suspend (via SYSCFG 40h[0]), so will RESET# since RESET# is derived from RSTDRV.
PWRGD	H26	I		**Power Good:** This input reflects the "wired-OR" status of the external reset switch and the power good status from the power supply.
OSC_14MHZ	E5	I		**Timer Oscillator Clock:** This is the main clock used by the internal 8254 timers. It is connected to a 14.31818MHz oscillator. OSC_14MHZ is a 5.0V tolerant input, even when its power plane is connected to 3.3.V as long as the 5VREF pins of FireStar are connected to +5.0V.
OSC32	C7	I		**32KHz Clock:** This signal is used as a 32KHz clock input. It is used for power management and is usually the only active clock when the system is in Suspend mode. OSC32 is a 5.0V tolerant input, even when its power plane is connected to 3.3.V as long as the 5VREF pins of FireStar are connected to +5.0V.
CPUCLKIN	M5	I		**Feedback input to Circuitry:** This input clock must be equivalent to, and in phase with, the clock going to the CPU. **Note:** This is a CMOS-level input and therefore it is imperative that the rise time on this signal is less than or equal to 2.5ns.

3.2.4 ISA Interface Signal Set

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
Interrupt Controller Interface				
IRQ1	AF18	I	PCIDV1 8Ah = 00h	**Interrupt Request 1:** Normally connected to the keyboard controller. IRQ1 is a 5.0V tolerant input, even when its power plane is connected to 3.3.V as long as the 5VREF pins of FireStar are connected to +5.0V.
PIO10		I/O (4mA)	PCIDV1 8Ah ≠ 00h	**Programmable Input/Output 10:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
IRQA/IRQ3	AC19	I		**Programmable Interrupt Request A / IRQ3:** This input defaults to IRQ3, however, it can be programmed to route onto any ISA or PCI interrupt through PCIDV1 B0h. IRQA/IRQ3 is a 5.0V tolerant input, even when its power plane is connected to 3.3.V as long as the 5VREF pins of FireStar are connected to +5.0V.
IRQB/IRQ4	AD19	I		**Programmable Interrupt Request B / IRQ4:** This input defaults to IRQ4, however, it can be programmed to route onto any ISA or PCI interrupt through PCIDV1 B1h. IRQB/IRQ4 is a 5.0V tolerant input, even when its power plane is connected to 3.3.V as long as the 5VREF pins of FireStar are connected to +5.0V.
IRQC/IRQ5	AE19	I		**Programmable Interrupt Request C / IRQ5:** This input defaults to IRQ5, however, it can be programmed to route onto any ISA or PCI interrupt through PCIDV1 B2h. IRQC/IRQ5 is a 5.0V tolerant input, even when its power plane is connected to 3.3.V as long as the 5VREF pins of FireStar are connected to +5.0V.
IRQD/IRQ6	AF19	I		**Programmable Interrupt Request D / IRQ6:** This input defaults to IRQ6, however, it can be programmed to route onto any ISA or PCI interrupt through PCIDV1 B3h. IRQD/IRQ6 is a 5.0V tolerant input, even when its power plane is connected to 3.3.V as long as the 5VREF pins of FireStar are connected to +5.0V.
IRQE/IRQ7	AD20	I		**Programmable Interrupt Request E / IRQ7:** This input defaults to IRQ7, however, it can be programmed to route onto any ISA or PCI interrupt through PCIDV1 B4h.

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
IRQ8#	AE20	I	PCIDV1 8Bh = 00h	**Interrupt Request 8:** Normally connected to the RTC alarm output.
PIO11		I/O (4mA)	PCIDV1 8Bh ≠ 00h	**Programmable Input/Output 11:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
IRQF/IRQ9	AF20	I		**Programmable Interrupt Request F / IRQ9:** This input defaults to IRQ9, however, it can be programmed to route onto any ISA or PCI interrupt through PCIDV1 B5h.
IRQG/IRQ10	AB22	I		**Programmable Interrupt Request G / IRQ10:** This input defaults to IRQ10, however, it can be programmed to route onto any ISA or PCI interrupt through PCIDV1 B6h.
IRQH/IRQ11	AC21	I		**Programmable Interrupt Request H / IRQ11:** This input defaults to IRQ11, however, it can be programmed to route onto any ISA or PCI interrupt through PCIDV1 B7h.
IRQ12	AD21	I	PCIDV1 8Ch = 00h	**Interrupt Request 12:** Normally connected to the mouse interrupt from the keyboard controller.
PIO12		I/O (4mA)	PCIDV1 8Ch ≠ 00h	**Programmable Input/Output 12:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
IRQ14	AE21	I	PCIDV1 8Dh = 00h	**Interrupt Request 14:** Normally connected to the primary IDE channel.
PIO13		I/O (4mA)	PCIDV1 8Dh ≠ 00h	**Programmable Input/Output 13:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
IRQ15	AF21	I	PCIDV1 BBh[0] = 0	**Interrupt Request 15:** Normally connected to the secondary IDE channel.
SIN#			PCIDV1 BBh[0] = 1	**Serial Input:** Serial interrupt return line for Intel style of serial IRQs.
IRQSER	AE18	I/O	PCIDV1 BAh[0] = 0	**Serial Interrupt Request:** Bidirectional interrupt line for Compaq style of serial IRQs.
SDCKE		O	PCIDV1 53h[4] = 1	**SDRAM Clock Enable:** This signal is asserted to put the SDRAM into a “Stop” state. The BIOS can program FireStar to assert this signal only in Suspend mode.
SOUT#			PCIDV1 BBh[0] = 1	**Serial Output:** Serial interrupt output line for Intel style of serial IRQs.

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
ISA DMA Arbiter Interface				
DRQA/DRQ0	M24	I	PCIDV1 99h = 00h	**Programmable DMA Request A / DRQ0:** The DRQ is used to request DMA service from the DMA controller. This input defaults to DRQ0, however, it can be programmed to route onto any internal DRQ by programming PCIDV1 C0h[2:0].
PIO25		I/O (4mA)	PCIDV1 99h ≠ 00h	**Programmable Input/Output 25:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
DRQB/DRQ1	M25	I	PCIDV1 9Ah = 00h	**Programmable DMA Request B / DRQ1:** The DRQ is used to request DMA service from the DMA controller. This input defaults to DRQ1, however, it can be programmed to route onto any internal DRQ by programming PCIDV1 C0h[6:4].
PIO26		I/O (4mA)	PCIDV1 9Ah ≠ 00h	**Programmable Input/Output 26:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
DRQC/DRQ2	M26	I	PCIDV1 9Bh = 00h	**Programmable DMA Request C / DRQ2:** The DRQ is used to request DMA service from the DMA controller. This input defaults to DRQ0, however, it can be programmed to route onto any internal DRQ by programming PCIDV1 C1h[2:0].
PIO27		I/O (4mA)	PCIDV1 9Bh ≠ 00h	**Programmable Input/Output 27:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
DRQD/DRQ3	L23	I	PCIDV1 9Ch = 00h	**Programmable DMA Request D / DRQ3:** The DRQ is used to request DMA service from the DMA controller. This input defaults to DRQ3, however, it can be programmed to route onto any internal DRQ by programming PCIDV1 C1h[6:4].
PIO28		I/O (4mA)	PCIDV1 9Ch ≠ 00h	**Programmable Input/Output 28:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
DRQE/DRQ5	L24	I	PCIDV1 9Dh = 00h	**Programmable DMA Request E / DRQ5:** The DRQ is used to request DMA service from the DMA controller. This input defaults to DRQ5, however, it can be programmed to route onto any internal DRQ by programming PCIDV1 C2h[6:4].
PIO29		I/O (4mA)	PCIDV1 9Dh ≠ 00h	**Programmable Input/Output 29:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
DRQF/DRQ6	L25	I	PCIDV1 9Eh = 00h	**Programmable DMA Request F / DRQ6:** The DRQ is used to request DMA service from the DMA controller. This input defaults to DRQ6, however, it can be programmed to route onto any internal DRQ by programming PCIDV1 C3h[2:0].
PIO30		I/O (4mA)	PCIDV1 9Eh ≠ 00h	**Programmable Input/Output 30:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
DRQG/DRQ7	L26	I	PCIDV1 9Fh = 00h	**Programmable DMA Request G / DRQ6:** The DRQ is used to request DMA service from the DMA controller. This input defaults to DRQ7, however, it can be programmed to route onto any internal DRQ by programming PCIDV1 C3h[6:4].
PIO31		I/O (4mA)	PCIDV1 9Fh ≠ 00h	**Programmable Input/Output 31:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
DACKA#/DACK0#	K22	O		**Programmable DMA Acknowledge A / DACK0#:** DACK# is used to acknowledge DRQ to allow DMA transfer. This input defaults to DACK0#, however, it can be programmed to route onto any internal DACK# by programming PCIDV1 C0h[2:0].
PPWR4			PCIDV1 C0h[2:0] = 100	**Peripheral Power Control Line 4:** Peripheral power control lines 0 through 15 are latch outputs used to control external devices.
DACKB#/DACK1#	K23	O		**Programmable DMA Acknowledge B / DACK1#:** DACK# is used to acknowledge DRQ to allow DMA transfer. This input defaults to DACK1#, however, it can be programmed to route onto any internal DACK# by programming PCIDV1 C0h[6:4].
PPWR5			PCIDV1 C0h[6:4] = 100	**Peripheral Power Control Line 5**
DACKC#/DACK2#	K24	O		**Programmable DMA Acknowledge C / DACK2#:** DACK# is used to acknowledge DRQ to allow DMA transfer. This input defaults to DACK2#, however, it can be programmed to route onto any internal DACK# by programming PCIDV1 C1h[2:0].
PPWR6			PCIDV1 C1h[2:0] = 100	**Peripheral Power Control Line 6**

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
DACKD#/DACK3#	K25	O		**Programmable DMA Acknowledge D / DACK3#:** DACK# is used to acknowledge DRQ to allow DMA transfer. This input defaults to DACK3#, however, it can be programmed to route onto any internal DACK# by programming PCIDV1 C1h[6:4].
PPWR7			PCIDV1 C1h[6:4] = 100	**Peripheral Power Control Line 7**
DACKE#/DACK5#	K26	O		**Programmable DMA Acknowledge E / DACK5#:** DACK# is used to acknowledge DRQ to allow DMA transfer. This input defaults to DACK5#, however, it can be programmed to route onto any internal DACK# by programming PCIDV1 C2h[6:4].
PPWR13			PCIDV1 C2h[6:4] = 100	**Peripheral Power Control Line 13**
DACKF#/DACK6#	J22	O		**Programmable DMA Acknowledge F / DACK6#:** DACK# is used to acknowledge DRQ to allow DMA transfer. This input defaults to DACK6#, however, it can be programmed to route onto any internal DACK# by programming PCIDV1 C3h[2:0].
PPWR14			PCIDV1 C3h[2:0] = 100	**Peripheral Power Control Line 14**
DACKG#/DACK7#	J23	O		**Programmable DMA Acknowledge G / DACK7#:** DACK# is used to acknowledge DRQ to allow DMA transfer. This input defaults to DACK7#, however, it can be programmed to route onto any internal DACK# by programming PCIDV1 C3h[6:4].
PPWR15			PCIDV1 C3h[6:4] = 100	**Peripheral Power Control Line 15**
Compact ISA Interface				
RSTDRV	AC25	I/O (4mA)	PCIDV1 8Fh = 00h	**Reset Drive:** Active high reset signal to ISA bus devices. RSTDRV can be programmed to toggle in Suspend via SYSCFG 40h[0].
PIO15			PCIDV1 8Fh ≠ 00h	**Programmable Input/Output 15:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
SD[15:0]	Refer to Table 3-2	I/O (8mA)	Cycle Multiplexed	**System Data Bus:** SD[15:0] provides the 16-bit data path for devices residing on the ISA bus.
MAD[15:0]				**Multiplexed Address/Data Bus:** Used during CISA cycles.

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
SEL/ATB#	AC20	I/O (4mA)	PCIDV1 8Eh = 00h	**Select/AT Back-off:** Dedicated CISA input. This signal needs to be pulled up externally.
SDCKE			PCIDV1 53h[3] = 1	**SDRAM Clock Enable:** This signal is asserted to put the SDRAM into a “Stop” state. The BIOS can program FireStar to assert this signal only in Suspend mode.
PIO14			PCIDV1 8Eh ≠ 00h	**Programmable Input/Output 14:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
CMD#	AB20	O (4mA)	SYSCFG 16h[7, 5]	**Command:** Dedicated CISA output used to signal a data transfer command.
DIRTY		I/O (4mA)		**Tag Dirty Bit:** This dirty bit allows the tag data to be 8 bits wide instead of 7.
PCICLK3		O (4mA)	ROMCS#: KBDCS# strap option	**PCI Clock Output 3**
ATCLK	AA22	O (8mA)		**ISA Bus Clock:** This signal is derived from an internal division of PCICLK. It is used to sample and drive all ISA synchronous signals. PCIDV1 47h[5:4] sets the ATCLK: 00 = PCICLK÷4 10 = PCICLK÷2 01 = PCICLK÷3 11 = PCICLK The ATCLK is also used to demultiplex and sample externally multiplexed inputs. During Suspend, it is possible to output 32KHz on this pin, or drive it low.
PCICLK4			ROMCS#: KBDCS# strap option	**PCI Clock Output 4**
IOCHRDY	AB26	I/O (8mA)		**I/O Channel Ready:** Resources on the ISA bus deassert IOCHRDY to indicate that wait states are required to complete the cycle. IOCHRDY is an input when FireStar owns the ISA bus and is an output when an external ISA bus master owns the ISA bus. IOCHRDY is automatically tristated in Suspend.
BALE	W22	O (8mA)		**Bus Address Latch Enable:** BALE is an active high signal asserted to indicate that the address, AEN, and SBHE# signal lines are valid. BALE remains asserted throughout ISA master and DMA cycles.
PCICLK5			ROMCS#: KBDCS# strap option	**PCI Clock Output 5**

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
ISA Bus Interface				
MRD#	AC26	I/O (8mA)		**Memory Read:** MRD# is the command to a memory slave that it may drive data onto the ISA data bus. MRD# is an output when FireStar is a master on the ISA bus. MRD# is an input when an ISA master, other than FireStar, owns the ISA bus.
IDE1_DCS3#			RTCAS:A20M# strap option	**DCS3 Control for Secondary IDE Channel**
MWR#	AB23	I/O (8mA)		**Memory Write:** MWR# is the command to a memory slave that it may latch data from the ISA data bus. MWR# is an output when the FireStar owns the ISA bus. MWR# is an input when an ISA master, other than FireStar, owns the ISA bus.
IDE1_DCS1#			RTCAS:A20M# strap option	**DCS1 Control for Secondary IDE Channel**
IOR#	AB24	I/O (8mA)		**I/O Read:** IOR# is the command to an ISA I/O slave device that the slave may drive data on to the ISA data bus (SD[15:0]). The I/O slave device must hold the data valid until after IOR# is negated. IOR# is an output when FireStar owns the ISA bus. IOR# is an input when an external ISA master owns the ISA bus.
IDE1_DRD#			RTCAS:A20M# strap option	**Drive Read Control for Secondary IDE Channel**
IOW#	AB25	I/O (8mA)		**I/O Write:** IOW# is the command to an ISA I/O slave device that the slave may latch data from the ISA data bus (SD[15:0]). IOW# is an output when FireStar owns the ISA bus. IOW# is an input when an external ISA master owns the ISA bus.
IDE1_DWR#			RTCAS:A20M# strap option	**D Write Control for Secondary IDE Channel**
SMRD#	W26	I/O (8mA)	PCIDV1 96h = 00h	**System Memory Read:** FireStar asserts SMRD# to request a memory slave to provide data. If the access is below the 1MB range (00000000h-000FFFFFh) during DMA compatible, IPC master, or ISA master cycles, FireStar asserts SMRD#.
PIO21			PCIDV1 96h ≠ 00h	**Programmable Input/Output 21:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
SMWR#	V22	I/O (8mA)	PCIDV1 97h = 00h	**System Memory Write:** FireStar asserts SMWR# to request a memory slave to accept data from the data lines. If the access is below the 1MB range (00000000h-000FFFFFh) during DMA compatible, IPC master, or ISA master cycles, FireStar asserts SMWR#.
PIO22			PCIDV1 97h ≠ 00h	**Programmable Input/Output 22:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
AEN	M22	I/O	PCIDV1 C2h[1] = 0	**Address Enable:** AEN is asserted during DMA cycles to prevent I/O slaves from misinterpreting DMA cycles as valid I/O cycles. When asserted, AEN indicates to an I/O resource on the ISA bus that a DMA transfer is occurring. This signal is asserted also during refresh cycles. AEN is driven low upon reset.
PPWR11			PCIDV1 C2h[1] = 1	**Peripheral Power Control Line 11**
IO16#	W23	I/O	PCIDV1 92h = 00h	**16-Bit I/O Chip Select:** This signal is driven by I/O devices on the ISA bus to indicate that they support 16-bit I/O bus cycles.
PIO18			PCIDV1 92h ≠ 00h	**Programmable Input/Output 18:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
M16#	W24	I/O	PCIDV1 93h = 00h	**16-Bit Memory Chip Select:** ISA slaves that are 16-bit memory devices drive this signal low. MEMCS16# is an input when FireStar owns the ISA bus. FireStar drives this signal low during ISA master to PCI memory cycles.
PIO19			PCIDV1 93h ≠ 00h	**Programmable Input/Output 19:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
RFSH#	J25	I/O	PCIDV1 C2h[0] = 0	**Refresh:** As an output, this signal is used to inform FireStar to refresh the local DRAM. During normal operation, a low pulse is generated every 15µs to indicate to FireStar that the DRAM is to be refreshed if PCIDV1 64h[0] = 0. During Suspend, if normal DRAM is used, the 32KHZ input to the FireStar is routed out on this pin so that it may perform DRAM refresh. An option to continuously drive this signal low during Suspend is also provided. The internal pull-up on this pin is disengaged in Suspend.
PPWR12			PCIDV1 C2h[0] = 1	**Peripheral Power Control Line 12**

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
SBHE#	W25	I/O	PCIDV1 94h = 00h	**System Byte High Enable:** When asserted, SBHE# indicates that a byte is being transferred on the upper byte (SD[15:8]) of the data bus. SBHE# is negated during refresh cycles. SBHE# is an output when FireStar owns the ISA bus.
PIO20			PCIDV1 94h ≠ 00h	**Programmable Input/Output 20:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
TC	M23	I/O	PCIDV1 C2h[2] = 0	**Terminal Count**
PPWR10			PCIDV1 C2h[2] = 1	**Peripheral Power Control Line 10**
XD7	AA23	I/O (8mA)	Cycle Multiplexed (See Note)	**XD Bus Line 7:** ISA status signal.
IDE_DCS3#				**DCS3 Control for Primary IDE Channel**
XD6	AA24	I/O (8mA)	Cycle Multiplexed (See Note)	**XD Bus Line 6:** ISA status signal.
IDE_DCS1#				**DCS1 Control for Primary IDE Channel**
XD5	AA25	I/O (8mA)	Cycle Multiplexed (See Note)	**XD Bus Line 5:** ISA status signal.
IDE_DDACK#				**DMA Acknowledge for Primary IDE Channel**
XD4	AA26	I/O (8mA)	Cycle Multiplexed (See Note)	**XD Bus Line 4:** ISA status signal.
IDE_DA2				**Address Bit 2 for Primary IDE Channel**
XD3	Y23	I/O (8mA)	Cycle Multiplexed (See Note)	**XD Bus Line 3:** ISA status signal.
IDE_DA1				**Address Bit 1 for Primary IDE Channel**
XD2	Y24	I/O (8mA)	Cycle Multiplexed (See Note)	**XD Bus Line 2:** ISA status signal.
IDE_DA0				**Address Bit 0 for Primary IDE Channel**
XD1	Y25	I/O (8mA)	Cycle Multiplexed (See Note)	**XD Bus Line 1:** ISA status signal.
IDE_DRD#				**Drive Read Control for Primary IDE Channel**
XD0	Y26	I/O (8mA)	Cycle Multiplexed (See Note)	**XD Bus Line 0:** ISA status signal.
IDE_DWR#				**Drive Write Control for Primary IDE Channel**
Note: XD[7:0] can be strapped to be dedicated IDE lines via the RTCAS:A20M# strap option and PCIDV1 75h[6] = 1.				
SA[23:20]	V23:V26,	I/O (8mA)		**System Address Bus Lines 23 through 20:** The SA[23:0] signals on FireStar provide the address for memory and I/O accesses on the ISA bus. The addresses are outputs when FireStar owns the ISA bus and are inputs when an external ISA master owns the ISA bus.
PPWR[3:0]			DBEW# strap option	**Peripheral Power Control Lines 3 through 0**

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
SA[19:18]	U23, U24	I/O (8mA)		**System Address Bus Lines 19 and 18**
PPWR[9:8]			DBEW# strap option	**Peripheral Power Control Lines 9 and 8**
SA[17:16]	U25, U26	I/O (8mA)	PCIDV1 91h-90h = 00h	**System Address Bus Lines 17 and 16**
PIO[17:16]			PCIDV1 91h-90h ≠ 00h	**Programmable Input/Output Lines 17 and 16:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
SA[15:0]	Refer to Table 3-2	I/O (8mA)		**System Address Bus Lines 15 through 0**
IDE1_DD[15:0]			RTCAS:A20M# strap option	**Disk Data Lines 15 through 0:** DD[15:0] provide the 16-bit data path for the IDE disk drives.
External Real-Time Clock Interface				
RTCAS	N24 Strap option pin, refer to Table 3-7	O (4mA)		**Real-Time Clock Address Strobe:** This signal is connected to the address strobe of the real-time clock.
IDE1_DA0		I/O	RTCAS:A20M# strap option and PCIDV1 75h[7] = 1.	**Address Bit 0 for Secondary IDE Channel**
RTCRD#	N25 Strap option pin, refer to Table 3-7	O (4mA)		**Real-Time Clock Read:** This pin is used to drive the read signal of the real-time clock.
IDE1_DA1		I/O	RTCAS:A20M# strap option and PCIDV1 75h[7] = 1.	**Address Bit 1 for Secondary IDE Channel**
RTCWR#	N26 Strap option pin, refer to Table 3-7	O (4mA)		**Real-Time Clock Write:** This pin is used to drive the write signal of the real-time clock.
IDE1_DA2		I/O	RTCAS:A20M# strap option and PCIDV1 75h[7] = 1.	**Address Bit 2 for Secondary IDE Channel**
Power Management Unit Interface				
PPWRL	AC23	O (4mA)	(Default)	**Power Control Latch:** This signal is used to control the external latching of the peripheral power control signals PPWR[15:0]. This signal is pulsed after reset to preset the external latch.
PPWR0#		I/O	BOFF# strap option	**Peripheral Power Control Line 0#**

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
Miscellaneous				
A20M#	R3 Strap option pin, refer to Table 3-7	O (4mA)		**Address Bit 20 Mask:** This pin is an output and generates the A20M# output by trapping GATEA20 commands to the keyboard or to Port 092h. The CPUINIT signal to the CPU is generated whenever it senses reset commands to Port 060h/064h, or a Port 092h write command with bit 0 set high. When keyboard emulation is disabled, FireStar traps only Port 092h GATEA20 commands and accepts the GATEA20 input from the keyboard controller, which is sent out as A20M# to the CPU.
ROMCS#	J24 Strap option pin, refer to Table 3-7	O (4mA)	PCIDV1 52h[2] = 0 97h = 00h 4Fh[1] = 0	**BIOS ROM Chip Select:** This output goes active on both reads and writes to the ROM area to support flash ROM. For flash ROM support, writes to ROM can be supported by appropriately setting PCIDV1 47h[7].
PIO23		I/O (4mA)	PCIDV1 52h[2] = 0 97h ≠ 00h 4Fh[1] = 0	**Programmable Input/Output Line 23:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
ROMCS#/ KBDCS#		O (4mA)	PCIDV1 52h[2] = 1 or 4Fh[1] = 1	**Combined ROM and Keyboard Chip Select:** When this combined functionality is selected, the ROM cycles are qualified by MRD#/MWR#; the keyboard controller cycles are qualified by IOR#/IOW#.
SPKROUT	H23	I/O (8mA)		**Speaker Data:** This pin is used to drive the system board speaker. This signal is a function of the Timer-0 Counter-2 and Port 061h bit 1. Can use CISA protocol to gang several.
KBDCS#	J26 Strap option pin, refer to Table 3-7	O (8mA)	Default PCIDV1 98h = 00h	**Keyboard Chip Select**: Used to decode accesses to the keyboard controller.
PIO24		I/O (8mA)	PCIDV1 98h ≠ 00h	**Programmable Input/Output 24:** See Section 3.3, "Programmable I/O Pins", on page 33 for more details.
DRD#		O (8mA)	PCIDV1 4Fh[1] = 1	**Drive Read Signal**

3.2.5 Test Mode Selection Pins

Signal Name	Pin No.	Signal Type (Drive)	Selected By	Signal Description
RSVD	B7 Strap option pin for future 2.5V CPU interface, refer to Table 3-7	I/O (4mA)		**Reserved:** This pin is reserved for possible additional functionality on future revisions of FireStar. However, it is used as an input for the ATE Test Mode selection address. See TMS (pin AB5) description.
RSVD	A7	I/O (4mA)		**Reserved in FireStar:** An input for the ATE Test Mode selection address. See TMS (pin AB5) description.
SDCKE (FS ACPI)			PCIDV1 52h[3] = 1	**SDRAM Clock Enable in FireStar ACPI:** This signal is asserted to put the SDRAM into a “Stop” state. The BIOS can program FireStar to assert this signal only in Suspend mode. This pin is also an input for the ATE Test Mode selection address. See TMS (pin AB5) description.
TMS	AB5	I/O		**Test Mode Select:** An input for the ATE Test Mode selection address. AB5 B7 A7 Mode 0 X X Normal operation (default) 1 0 0 Tristate all pins 1 0 1 NAND tree test 1 1 0 Reserved for factory test 1 1 1 Reserved for factory test

3.2.6 Power and Ground Pins

Signal Name	Pin No.	Signal Type	Signal Description
GND	AA6, AA13, AA14, AA21, AB13, E14, F6, F13, F14, F21, N5, N6, N21, P6, P21, P22	G	**Ground Connections**
VCC_ISA	L22, U22, Y22	P	**ISA Bus Power Plane: 3.3V or 5.0V**
VCC_CPU	E8, G5, T5, W5	P	**CPU Bus Power Plane: 3.3V (and 2.5V in future 2.5V CPU interface revision)**
VCC_CORE	AB19, H22, K5	P	**FireStar Core Power Plane: 3.3V only**
VCC_DRAM	E11, E17, E20	P	**Memory Power Plane: 3.3V or 5.0V**
VCC_PCI	AB7, AB10, AB16	P	**PCI Bus Power Plane: 3.3V or 5.0V**
5VREF	AB21, E7	P	**5.0 V Reference:** Connect to 5.0V is available in the system. Connect to 3.3V for an all 3.3V design.

3.3 Programmable I/O Pins

Programmable I/O options are available on 32 PIO pins of the FireStar chip. The options available comprise all of the PPWR[15:0] power control lines, the general purpose chip select lines GPCS[3:0]#, certain ISA pins, and various other options such as logical operations (OR, AND, NOT, XOR). The goal is to reduce to a minimum the number of external logic devices required by recovering unused pins.

PIO pin assignment is more flexible than in past OPTi chips, allowing any programmable function to replace any PIO-ready pin. This assignment always overrides the former function of the pin. The mechanism for PIO assignment is to program the group and subfunction number for the PIO pin into the corresponding register at PCIDV1 80h-9Fh.

Functions are also assignable to internal "nodes". In this way, multi-level logic functions can be built up internally and the resulting input or output signals can be assigned to free PIO pins.

Note: Any signal that can potentially be programmed as a PIO pin will function as a PIO pin only if the corresponding CPU register is programmed to a non-zero value.

Table 3-3 PIO Functions

Group	Function	Sub-function Number	Description
Power Management Inputs Group 0	Default on pin	0h	Pin definition at reset
	EPMI0#	1h	External Power Management Input 0
	EPMI1#	2h	External Power Management Input 1
	EPMI2#	3h	External Power Management Input 2
	EPMI3#	4h	External Power Management Input 3
	LOBAT	5h	Low Battery SMI (periodic)
	LLOBAT	6h	Very Low Battery SMI (level-triggered)
	RINGI	7h	Ring Indicator
	SUS/RES#	8h	Suspend/Resume
	THMIN	9h	
	HDI	Ah	ISA Hot Docking Indicator
	TEMPDET	Bh	Temperature Detect Input for thermal mgmt.
	Reserved	C-Fh	
Power Control Outputs Group 1h	PPWRx	0-Fh	Peripheral Power Control Outputs, x = 0..15
Misc. Inputs Group 2h	PCIRQ[3:0]#	0-3h	PCI Interrupts
	DDRQ1	4h	IDE Cable 1 DMA Request
	CHRDYA	5h	Dedicated IDE Cable 0 Channel Ready
	CHRDYB	6h	Dedicated IDE Cable 1 Channel Ready
	MSTR#	7h	ISA MASTER# signal
	CHCK#	8h	ISA IOCHCK# signal (generates NMI)
	KBCRST	9h	Reset signal from Keyboard Controller
	KBCA20M#	Ah	A20M# signal from Keyboard Controller
	Monitor Input	Bh	PIO pin becomes input; read at PCIDV1 A8h-ABh
	NOWS#	Ch	ISA zero wait state signal
	Reserved	D-Fh	

Table 3-3 PIO Functions (cont.)

Group	Function	Sub-function Number	Description
Misc. Outputs Group 3h	GPCSx#	0-3h	General Purpose Chip Select outputs, x=0-3
	Reserved	4-7h	
	CDIR	8h	Compact ISA Cable Buffer Direction signal
	L2CLKOE	9h	L2 Cache Clock Output Enable
	PCICLKOE	Ah	PCI Clock Output Enable (to ext. clock generator)
	HGNT#	Bh	UMA Split Buffer Control signal
	FAN	Ch	CPU overtemp fan control output
	Reserved	Dh	
	PCTLL	Eh	Power control latch low is available only on PIO15 (RSTDRV) and is used to latch PPWR[15:0] from SD[15:0]
	ATCLK/2	Fh	ATCLK divided by 2 (for KBCLK and/or ACPIMX)
IDE Controller Outputs Group 4h	DDACK0#	0h	Dedicated IDE DMA acknowledge (Primary cable)
	DDACK1#	1h	Dedicated IDE DMA acknowledge (Secondary cable)
	DRD#	2h	Dedicated IDE command
	DWR#	3h	
	DCS1#	4h	Dedicated IDE chip select
	DCS3#	5h	
	DA0	6h	Dedicated IDE address
	DA1	7h	
	DA2	8h	
	DBEX#	9h	IDE buffer control for drive X
	DBEY#	Ah	IDE buffer control for drive Y
	DBEZ#	Bh	IDE buffer control for drive Z
	DDACK0-0#	Ch	Dedicated IDE DMA acknowledge (Primary Cable, Drive 0)
	DDACK0-1#	Dh	Dedicated IDE DMA acknowledge (Primary Cable, Drive 1)
	DDACK1-0#	Eh	Dedicated IDE DMA acknowledge (Secondary Cable, Drive 0)
	DDACK1-1#	Fh	Dedicated IDE DMA acknowledge (Secondary Cable, Drive 1)

Table 3-3 PIO Functions (cont.)

Group	Function	Sub-function Number	Description
ACPI Inputs Group 7h	UNDOCK# (ACPI0)	0h	Active low input
	DOCK# (ACPI1)	1h	Active low input
	STSCH# (ACPI2)	2h	Active low input
	FRI# (ACPI3)	3h	Active low input
	RI# (ACPI4)	4h	Active low input
	USB# (ACPI5)	5h	Active low input
	EC# (ACPI6)	6h	Active low input
	LID (ACPI7)	7h	Active high input
	(ACPI8)	8h	Active high input
	(ACPI9)	9h	Active high input
	(ACPI10)	Ah	Active high input
	(ACPI11)	Bh	Active high input
	ACPIMX0	Ch	Time-multiplexed input of ACPI0-3
	ACPIMX1	Dh	Time-multiplexed input of ACPI4-7
	ACPIMX2	Eh	Time-multiplexed input of ACPI8-11
	PWRBTN#	Fh	Power button with hardware-enforced Suspend feature

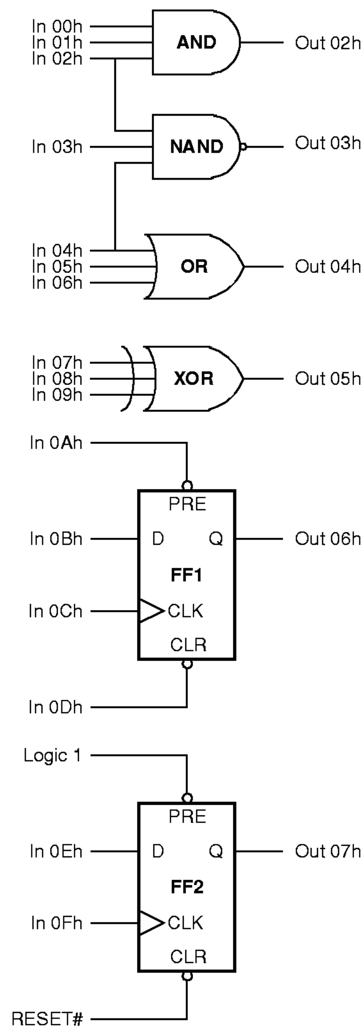
The PIO functions listed in Table 3-4 are simple gate functions. The available gates are illustrated in Figure 3-2.

Some PIO pins can be assigned a new signal function through the register set as shown in Table 3-6.

Table 3-4 Gate Level PIO Functions

Group	Function	Sub-function Number	Description
Gate Logic Inputs Group 5h	AND input 1	0h	3-input AND Gate
	AND input 2	1h	
	AND input 3/ NAND input 1	2h	
	NAND input 2	3h	3-input NAND Gate
	NAND input 3/ OR input 1	4h	3-input OR Gate
	OR input 2	5h	
	OR input 3	6h	
	XOR input 1	7h	3-input XOR Gate
	XOR input 2	8h	
	XOR input 3	9h	
	FF1 PRE# input	Ah	First D Flip-Flop
	FF1 D input	Bh	
	FF1 CLK input	Ch	
	FF1 CLR# input	Dh	
	FF2 D input	Eh	Second D Flip-Flop
	FF2 CLK input	Fh	
Logic Outputs Group 6h	Logic 0	0h	
	Logic 1	1h	
	AND output	2h	
	NAND output	3h	
	OR output	4h	
	XOR output	5h	
	FF1 Q output	6h	
	FF2 Q output	7h	
	Reserved	8-Fh	

Figure 3-2 Programmable Logic Matrix



The gate inputs and outputs can be connected directly to PIO pins, or can be connected to each other for multi-level logic development as described in the example below and using the Gate Matrix registers shown in Table 3-5.

Example:
A certain system design might require a nearly complete ISA bus, but without the need for the M16# pin because no ISA memory would be supported. PPWR6 function could be assigned to replace the M16# pin without disturbing the rest of the ISA interface by simply programming. PCIDV1 93h = 16h (M16# is PIO19). A setting of 16h selects the Power Control Outputs Group (1h) and the PPWR6 subfunction (6h).

Table 3-5 Gate Matrix Programming Registers

7	6	5	4	3	2	1	0
PCIDV1 A0h			**Logic Matrix Register 1**				**Default = 00h**
Invert input 01h (whether from PIO pin or from logic matrix output)? 0 = No 1 = Yes	Connect logic input 01h (AND2) to: 000 = PIO pin 001 = Logic 1 010 = Out 2h (AND output) 011 = Out 3h (NAND output) 100 = Out 4h (OR output) 101 = Out 5h (XOR output) 110 = Out 6h (flip-flop 1 output) 111 = Out 7h (flip-flop 2 output)			Invert input 00h (whether from PIO pin or from logic matrix output)? 0 = No 1 = Yes	Connect logic input 00h (AND1) to: Refer to PCIDV1 A0h[6:4] for decode.		
PCIDV1 A1h			**Logic Matrix Register 2**				**Default = 00h**
Invert input 03h? 0 = No 1 = Yes	Connect logic input 03h (NAND) to: Refer to PCIDV1 A0h[6:4] for decode.			Invert input 02h? 0 = No 1 = Yes	Connect logic input 02h (AND3) to: Refer to PCIDV1 A0h[6:4] for decode.		
PCIDV1 A2h			**Logic Matrix Register 3**				**Default = 00h**
Invert input 05h? 0 = No 1 = Yes	Connect logic input 05h (OR2) to: Refer to PCIDV1 A0h[6:4] for decode.			Invert input 04h? 0 = No 1 = Yes	Connect logic input 04h (OR1) to: Refer to PCIDV1 A0h[6:4] for decode.		
PCIDV1 A3h			**Logic Matrix Register 4**				**Default = 00h**
Invert input 07h? 0 = No 1 = Yes	Connect logic input 07h (XOR1) to: Refer to PCIDV1 A0h[6:4] for decode.			Invert input 06h? 0 = No 1 = Yes	Connect logic input 06h (OR3) to: Refer to PCIDV1 A0h[6:4] for decode.		
PCIDV1 A4h			**Logic Matrix Register 5**				**Default = 00h**
Invert input 09h? 0 = No 1 = Yes	Connect logic input 09h (XOR3) to: Refer to PCIDV1 A0h[6:4] for decode.			Invert input 08h? 0 = No 1 = Yes	Connect logic input 08h (XOR2) to: Refer to PCIDV1 A0h[6:4] for decode.		
PCIDV1 A5h			**Logic Matrix Register 6**				**Default = 00h**
Invert input 0Bh? 0 = No 1 = Yes	Connect logic input 0Bh (flip-flop 1, -D input) to: Refer to PCIDV1 A0h[6:4] for decode.			Invert input 0Ah? 0 = No 1 = Yes	Connect logic input 0Ah (flip-flop 1, PRE# input) to: Refer to PCIDV1 A0h[6:4] for decode.		
PCIDV1 A6h			**Logic Matrix Register 7**				**Default = 00h**
Invert input 0Dh? 0 = No 1 = Yes	Connect logic input 0Dh (flip-flop 1, CLR# input) to: Refer to PCIDV1 A0h[6:4] for decode.			Invert input 0Ch? 0 = No 1 = Yes	Connect logic input 0Ch (flip-flop 1, CLK input) to: Refer to PCIDV1 A0h[6:4] for decode.		
PCIDV1 A7h			**Logic Matrix Register 8**				**Default = 00h**
Invert input 0Fh? 0 = No 1 = Yes	Connect logic input 0Fh (flip-flop 2, CLK input) to: Refer to PCIDV1 A0h[6:4] for decode.			Invert input 0Eh? 0 = No 1 = Yes	Connect logic input 0Eh (flip-flop 1, D input) to: Refer to PCIDV1 A0h[6:4] for decode.		

Table 3-6 Register Programmable PIO Pins

7	6	5	4	3	2	1	0
PCIDV1 80h			**PIO0 Pin (CDOE#) Function Register**				**Default = 00h**
Tristate, pull-down PIO pin during Suspend: 0 = No 1 = Yes	000 = Group 0 (Power Management Inputs) 001 = Group 1 (Power Control Outputs) 010 = Group 2 (Miscellaneous Outputs) 011 = Group 3 (Miscellaneous Outputs) 100 = Group 4 (IDE Controller Outputs) 101 = Group 5 (Gate Logic Inputs) 110 = Group 6 (Logic Outputs) 111 = Group 7 (Reserved)			0000 = Group sub-function 0 0001 = Group sub-function 1 0010 = Group sub-function 2 0011 = Group sub-function 3 0100 = Group sub-function 4 0101 = Group sub-function 5 0110 = Group sub-function 6 0111 = Group sub-function 7		1000 = Group sub-function 8 1001 = Group sub-function 9 1010 = Group sub-function 10 1011 = Group sub-function 11 1100 = Group sub-function 12 1101 = Group sub-function 13 1110 = Group sub-function 14 1111 = Group sub-function 15	
PCIDV1 81h			**PIO1 Pin (TAGWE#) Function Register**				**Default = 00h**
PCIDV1 82h			**PIO2 Pin (ADSC#) Function Register**				**Default = 00h**
PCIDV1 83h			**PIO3 Pin (ADV#) Function Register**				**Default = 00h**
PCIDV1 84h			**PIO4 Pin (RAS2#) Function Register**				**Default = 00h**
PCIDV1 85h			**PIO5 Pin (RAS1#) Function Register**				**Default = 00h**
PCIDV1 86h			**PIO6 Pin (CLKRUN#) Function Register**				**Default = 00h**
PCIDV1 87h			**PIO7 Pin (REQ1#) Function Register**				**Default = 00h**
PCIDV1 88h			**PIO8 Pin (REQ2#) Function Register**				**Default = 00h**
PCIDV1 89h			**PIO9 Pin (DDRQ0) Function Register**				**Default = 00h**
PCIDV1 8Ah			**PIO10 Pin (IRQ1) Function Register**				**Default = 00h**
PCIDV1 8Bh			**PIO11 Pin (IRQ8#) Function Register**				**Default = 00h**
PCIDV1 8Ch			**PIO12 Pin (IRQ12) Function Register**				**Default = 00h**
PCIDV1 8Dh			**PIO13 Pin (IRQ14) Function Register**				**Default = 00h**
PCIDV1 8Eh			**PIO14 Pin (SEL#/ATB#) Function Register**				**Default = 00h**
PCIDV1 8Fh			**PIO15 Pin (RSTDRV) Function Register**				**Default = 00h**
PCIDV1 90h			**PIO16 Pin (SA16) Function Register**				**Default = 00h**
PCIDV1 91h			**PIO17 Pin (SA17) Function Register**				**Default = 00h**
PCIDV1 92h			**PIO18 Pin (IO16#) Function Register**				**Default = 00h**
PCIDV1 93h			**PIO19 Pin (M16#) Function Register**				**Default = 00h**
PCIDV1 94h			**PIO20 Pin (SBHE#) Function Register**				**Default = 00h**
PCIDV1 95h			**PIO21 Pin (SMRD#) Function Register**				**Default = 00h**
PCIDV1 96h			**PIO22 Pin (SMWR#) Function Register**				**Default = 00h**
PCIDV1 97h			**PIO23 Pin (ROMCS#) Function Register**				**Default = 00h**
PCIDV1 98h			**PIO24 Pin (KBDCS#) Function Register**				**Default = 00h**
PCIDV1 99h			**PIO25 Pin (DRQA) Function Register**				**Default = 00h**
PCIDV1 9Ah			**PIO26 Pin (DRQB) Function Register**				**Default = 00h**
PCIDV1 9Bh			**PIO27 Pin (DRQC) Function Register**				**Default = 00h**
PCIDV1 9Ch			**PIO28 Pin (DRQD) Function Register**				**Default = 00h**
PCIDV1 9Dh			**PIO29 Pin (DRQE) Function Register**				**Default = 00h**
PCIDV1 9Eh			**PIO30 Pin (DRQF) Function Register**				**Default = 00h**
PCIDV1 9Fh			**PIO31 Pin (DRQG) Function Register**				**Default = 00h**

3.4 Strap Selected Options

The strap options are selected by connecting a 10kohm resistor opposite to the sense of the internal resistor listed. The internal resistor is about 50kohm, so the external resistor must be less than 10kohm to counteract it.

Table 3-7 gives the strap options and Table 3-8 shows the registers used to readback the option selected.

Table 3-7 Strap Options

Pin No.	Pin Name	Option	Internally at Reset	Description
N25	RTCRD#	PCICLK1 Enable PCIDV1 48h[0]	Pull low (becomes GNT1# by default)	Selects whether: -GNT1# comes out (if RTCRD# is sensed low) -PCICLK1 comes out (if RTCRD# is sensed high)
N26	RTCWR#	PCICLK2 Enable PCIDV1 48h[1]	Pull low (becomes GNT2# by default)	Selects whether: -GNT2# comes out (if RTCWR# is sensed low) -PCICLK2 comes out (if RTCWR# is sensed high)
J24:J26	Bit 1:0 ROMCS#: KBDCS#	PCICLK3-5 Enable PCIDV1 48h[3:2]	Pull low (all pins take on default ISA/CISA function)	00 = CMD# is CMD#, ATCLK is ATCLK, BALE is BALE 01 = CMD# => PCICLK3, ATCLK and BALE stay the same 10 = CMD# => PCICLK3, ATCLK => PCICLK4, BALE stays the same 11 = CMD# => PCICLK3, ATCLK => PCICLK4, BALE => PCICLK5
AF5	INTR	PCIVCC Select PCIDV1 48h[4]	Pull high (PCI is 3.3V by default)	Selects whether input threshold on PCI interface is: -3.3V (if INTR is sensed high) -5.0V (if INTR is sensed low)
AD5	NMI	DRAMVCC Select PCIDV1 48h[5]	Pull high (DRAM is 3.3V by default)	Selects whether input threshold on DRAM is: -3.3V (if NMI is sensed high) -5.0V (if NMI is sensed low)
AC6	IGERR#	ISAVCC Select PCIDV1 48h[6]	Pull low (ISA is 5.0V by default)	Selects whether input threshold on ISA interface is: -3.3V (if IGERR# is sensed high) -5.0V (if IGERR# is sensed low)
H24	DBEW#	PPWR Select PCIDV1 49h[1]	Pull low (Normal mode is selected by default)	Force SA[23:18] to zero during reset to use as initially low PPWR pins: -Normal mode (if DBEW# is sensed low) -Initially low PPWR[3:0] and PPWR[9:8] (if DBEW# is sensed high)
R5	BOFF#	PPWR0# Select PCIDV1 49h[2]	Pull low (becomes PPWRL by default)	Selects whether: -PPWRL comes out (if line is sensed low) -PPWR0# comes out (if line is sensed high)
N24:R3	Bit 1:0 RTCAS: A20M#	Mode Select PCIDV1 49h[3,0]	Pull high (Normal decode, ISA-less mode is selected by default)	Selects whether: 00 = PC98 Mode, ISA-less Mode 01 = Normal decode ISA mode. ROM, KBC, and RTC on SD bus only (XD bus is not sampled during XD bus device read cycles). The XD bus is automatically mapped as dedicated primary channel IDE control. PCIDV1 46h[6] is ignored. 10 = Normal decode ISA mode, ROM on XD bus only. KBC and RTC can be relocated to the SD bus. XD bus can be qualified with DBE# to generate IDE control signals. The XD bus data will also be driven onto the SD bus during XD bus device read cycles. 11 = Normal decode, ISA-less Mode
AB14	PCICLK0	MCACHE Support PCIDV1 49h[4]	Pull low (No MCACHE support)	Selects whether: 0 = No MCACHE support 1 = MCACHE support enabled (Feature is not supported)
B7	RSVD	CPUVCC	Pull low (CPU is 3.3V by default)	Selects whether input threshold on CPU interface is: -3.3V (if RSVD is sensed low) -2.5V (if RSVD is sensed high)

Table 3-8 Strap Option Readback Registers

7	6	5	4	3	2	1	0
PCIDV1 48h			**Strap Option Readback Register (RO) - Byte 0**				**Default = 00h**
Reserved	IGERR# strap option selects: 0 = 5.0V ISA 1 = 3.3V ISA	NMI strap option selects: 0 = 5.0V DRAM 1 = 3.3V DRAM	INTR strap option selects: 0 = 5.0V PCI 1 = 3.3V PCI	ROMCS#:KBDCS# strap option selects: 00 = CMD#, ATCLK, BALE 01 = PCICLK3, ATCLK, BALE 10 = PCICLK3, PCICLK4, BALE 11 = PCICLK3, PCICLK4, PCICLK5		RTCWR# strap option selects: 0 = GNT2# 1 = PCICLK2	RTCRD# strap option selects: 0 = GNT1# 1 = PCICLK1
PCIDV1 49h			**Strap Option Readback Register (RO) - Byte 1**				**Default = 00h**
Reserved			PCICLK0 strap option selects: 0 = No MCACHE 1 = MCACHE enabled Not supported.	RTCAS strap selects(1)	BOFF# strap option selects: 0 = PPWRL 1 = PPWR0#	DBEW# strap option selects: 0 = SA[23:18] pins are SA[23:8] signals 1 = SA[23:18] pins are remapped: SA[23:20] = PPWR[3:0] and SA[19:18] = PPWR[9:8]	A20M# strap selects(1)
(1) Bits 3 and 0 work together: 00 = NEC mode & No ISA mode 01 = ISA mode without XD bus 10 = ISA mode with XD bus 11 = No ISA mode							

4.0 Functional Description

4.1 Buses and Concurrency

The FireStar architecture is built around multiple system buses. These are independent buses except for ISA/Compact ISA, IDE, and the X bus, which all share at least part of their pins. The buses are described below in the order of decode priority. If no priority is listed within a group, then there is no implied priority (no conflicts among participating devices).

4.1.1 Cycles Originating from CPU

Cycles started by the CPU go first to DRAM and/or L2 cache, then to PCI. From PCI the cycle can be:

- Positively decoded by the FireStar logic itself, in the case of IDE or known ISA devices.
- Positively decoded by a local device on the PCI bus.
- Positively decoded by a bridge chip such as the 82C824 CardBus/Docking Controller, subject to later rejection if no downstream device claims the cycle (the aborted cycle eventually gets claimed by the local ISA bus).
- Subtractively decoded and forwarded to ISA/CISA.

The highest priority in any CPU-initiated cycle will always be to the locally decoded L2 cache and DRAM. The ranges decoded:

Logic	Memory Ranges Decoded
CPU/L2 cache	0-128MB
DRAM	0-512MB

If there is no decode by local memory, the cycle goes out on the PCI bus. There are three possible cases of how the FireStar system handles the cycle on PCI, according to its intended destination:

- Cycles destined for a known local device.
- Cycles destined for unknown device on PCI, local Compact ISA, or local ISA.
- Cycles destined for a docking station ISA bus device.

4.1.1.1 Cycles Destined for a Known Local Device

For access to all devices known to FireStar, such as local IDE, ROM, RTC, local ISA floppy, etc., the PCI cycle is automatically remapped to an address space in high memory (a "base address" register is provided in the PCI configuration register space). The full list of positively decoded devices is provided in Section 4.9.4.1, "PCI Positive Decode for ISA". In this way, PCI bus devices will not attempt to claim the cycle. FireStar always responds to this cycle with a fast PCI decode.

PCI bus masters other than FireStar do not need to issue a remapped address to access local devices. In this case, FireStar must wait for the subtractive decode clock before claiming the cycle.

Table 4-1 indicates the cycles considered for positive decode/remapping, ordered by priority.

Table 4-1 Cycle Decode

Bus/Device	Memory Address Ranges	I/O Address Ranges	Remapped	Decode
X	C0000-FFFFFh	060, 064, 070-1h	MemBase or IOBase + original address	Positive by FireStar
IDE (PIO mode)	--	1F0-7, 3F6-7, 170-7, 376-7h	IOBase + original address	Positive by FireStar
Known local ISA devices	Defined by PMI decode	Defined by PMI decode	MemBase or IOBase + original address	Positive by FireStar
PCI	All	All	No	Positive by PCI device
Docking Station ISA	All	All	No	Positive by PCI device
Unknown local ISA devices, Compact ISA	All non-local DRAM space, or DRAM "holes"	x100-x3FFh	No	Subtractive by FireStar

4.1.1.2 Cycles Destined for Unknown Devices on PCI, Local Compact ISA, or Local ISA

For accesses whose owner is not known beforehand to the FireStar logic, the cycle presented on PCI is the same one as generated by the CPU. FireStar monitors DEVSEL#: If no PCI device claims the cycle, FireStar claims it at the subtractive decode clock and passes it to the local ISA controller. Local ISA always claims unclaimed cycles. Even if no ISA device is present to receive the cycle, PCI will see a normal cycle termination.

4.1.1.3 Cycles Destined for a Docking Station ISA Bus Device

Cycles that might be destined for docking station ISA will be claimed on the medium decode clock by the 82C824 chip. If the 82C825 PCI-ISA bridge cannot complete the cycle (no ISA device responds), FireStar logic must have some means of recovering the cycle.

Therefore, the 82C824 logic immediately and automatically retries all cycles to ISA windows (programmable on a per-window basis in the 82C824 registers) while it attempts to complete the access through the 82C825 bridge. The 82C824 continues to claim all retries while it awaits a response from the 82C825 chip.

Two outcomes to the PCI cycle on bus 0 are possible.

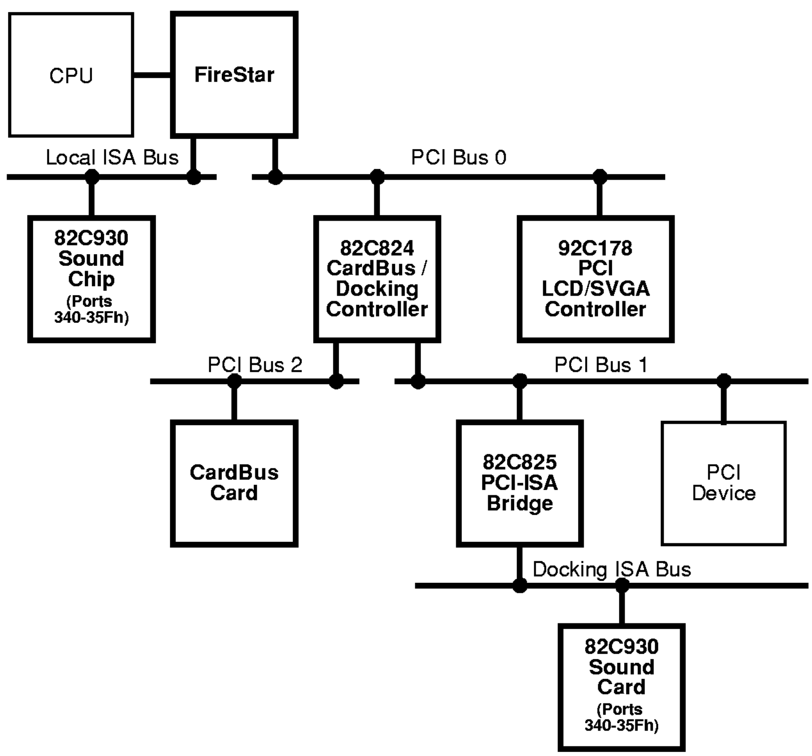
1) If the attempt to complete a cycle on the 82C825 ISA bus is successful, the 82C824 completes the cycle with the primary PCI bus master (FireStar or other master) on a later retry. FireStar will retry such cycles up to a programmable limit. Refer to Section 4.9.4.2, "Remote ISA Support" for details on how the FireStar/82C824/82C825 solution recognizes "claimed" cycles on ISA.

2) If the 82C825 fails to determine that an ISA device has claimed the cycle, it will end its PCI cycle with a Target Abort. As a result, the 82C824 chip, which has been forcing retry attempts on the primary side up until now, will simply ignore the next retry. In this way, FireStar will claim the cycle through subtractive decoding and pass it to its local ISA bus.

Figure 4-1 illustrates the multi-bus concept needed when using ISA on a docking station.

FireStar implements a register to limit the number of retries. In this way, a system hang condition can be resolved by allowing SMM or other code to interrogate the 82C824 chip and determine why the cycle cannot proceed. For example, if a bad docking connection has caused the 82C824 to be unable to properly complete its cycle, FireStar can still recover gracefully on the primary side after its retry limit has been reached.

Figure 4-1 Multiple ISA Bus Support



4.1.2 Concurrent Bus Operation

The independent nature of the buses allows all common bus operations to run concurrently. The only concurrency restrictions involve the low-speed slave buses, as follows.

- PIO mode IDE cycles effectively block all system buses if the cycles are initiated by the CPU, since the I/O cycle must first go through PCI. Bus master mode IDE cycles do not block the CPU.
- BIOS ROM cycles on the X bus block the IDE bus because they use the IDE control bus.
- RTC and keyboard controller cycles on the X bus block only IDE operations because they use the IDE control bus.
- ISA cycles block IDE cycles because they use the IDE data bus.

Since the BIOS is usually shadowed in DRAM, it will only be accessed at system boot time. Therefore, it is reasonable to summarize that only ISA cycles to unknown (non-positively decoded) devices will significantly reduce system performance. System designs should avoid integrating ISA bus devices where possible in order to allow maximum concurrency.

4.2 Intermodule Communications

The operations of the FireStar chip are specified in terms of communication sequences between logic modules. For example, a CPU write to DRAM can be broken down into two intermodule communication sequences: CPU to post-write buffers, then post-write buffers to DRAM.

The following sub-sections highlight specific aspects of intermodule communications in FireStar.

4.2.1 Read: CPU < DRAM or L2 Cache

Memory read cycles on the CPU interface are always directed to both the L2 cache controller and to the DRAM interface, in parallel. The memory controller always starts a memory read cycle to DRAM while it is waiting for the L2 cache tag comparison results, as long as the DRAM is not busy (such as for a PCI bus master access to DRAM). If the data turns out to be in L2 cache, the memory controller simply terminates the DRAM cycle.

Starting the cycle early allows the DRAM controller to generate an address on the MA lines and drop its RAS# line in preparation for the possible access to come.

4.2.2 Write: CPU > DRAM or L2 Cache

Any write to memory, whether destined for DRAM or PCI, is first posted to the post write buffers. The CPU interface logic controls the burst at its fastest speed, and can generate NA# at the appropriate time to pipeline the succeeding burst. The chip can accommodate six qwords (quadwords - 64-bit data words) for full-speed (no wait state) pipelined burst cycles even at 66MHz.

Once the post-write buffer is filled up, the CPU interface logic inserts CPU wait states to prevent further writes until some of the DRAM writes are completed. However, the CPU will be allowed to continue writing, one burst at a time, as the buffer empties (non-blocking feature).

4.2.3 Cache Write Hit - Write Cycles from CPU to L2 Cache

The FireStar solution minimizes the penalty of eventual write-back cycles to DRAM by using the OPTi-proprietary adaptive writeback scheme. In adaptive writeback mode, CPU writes to memory are automatically written-through to DRAM instead of being written only to L2 cache for a later writeback to DRAM. However, this action occurs only if a specified number of write buffers is available (programmable).

If the write buffers are filled above the limit (programmable), the system enters normal writeback mode and only updates the L2 cache. It will write the data back to DRAM only when that cache line is replaced in L2 cache.

4.2.4 L2 Cache Inquiries from PCI

PCI bus memory transactions are automatically presented on the CPU bus to determine whether the data exists in L2 cache. If so:

- For a PCI memory write cycle, the cache line will be invalidated.
- For a PCI memory read cycle in the local DRAM range, the cache line will be written back to the DRAM while the PCI read cycle is taking place. In this case, the PCI read cycle takes place concurrently with the writeback cycle to DRAM.

This mode of operation optimizes operations on PCI that involve DRAM.

4.3 Reset Logic

The PWRGD input is used to generate the CPU and the system reset, RESET#. PWRGD is a "cold reset" which is generated when either PWRGD goes low (from the power supply, indicating a low power condition) or the system reset button is activated. When PWRGD makes a low-to-high transition, RESET# will go active and will remain active for at least 1ms after PWRGD goes high. The PCI clock input to FireStar is used for timing the RESET# pulse width and must be running during reset. The RESET# output may be used to generate reset for all system devices. In addition, FireStar generates reset on the RSMRST signal which may be used to reset devices on power-up and also on a Resume from Suspend. The duration of the RSMRST pulse on power-up is fixed and is the same as the duration of the RESET# signal. However, the duration of the RSMRST pulse on Resuming from Suspend can be programmed as 8ms, 32ms, 128ms, 32 µs, or 1s. Refer to Table 4-2.

The CPUINIT signal is used to initialize the 3.3V CPU during warm resets. CPUINIT is generated for the following cases:

- When a shutdown condition is decoded from the CPU bus definition signals, the 82C700 will assert CPUINIT for 15 T-states.
- Keyboard reset to I/O Port 064h.
- Fast reset to I/O Port 092h.

Table 4-2 Resume Recovery Control Bits

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">SYSCFG 68h</td><td colspan="4">Clock Source Register 3</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2"></td><td colspan="2"></td><td colspan="2">Resume recovery time:
00 = 8ms 10 = 128ms
01 = 32ms 11 = 30µs
Note: Ignored if BEh[0] = 1.</td><td colspan="2"></td></tr>
<tr><td colspan="3">SYSCFG BEh if AEh[7] = 0</td><td colspan="3">Idle Reload Event Enable Register 2</td><td colspan="2">Default = 00h</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>Override SYSCFG 68h[3:2]:
0 = No
1 = Recover time 1s</td></tr>
</table>

4.4 System Clocks

4.4.1 CPU and FireStar Clocks

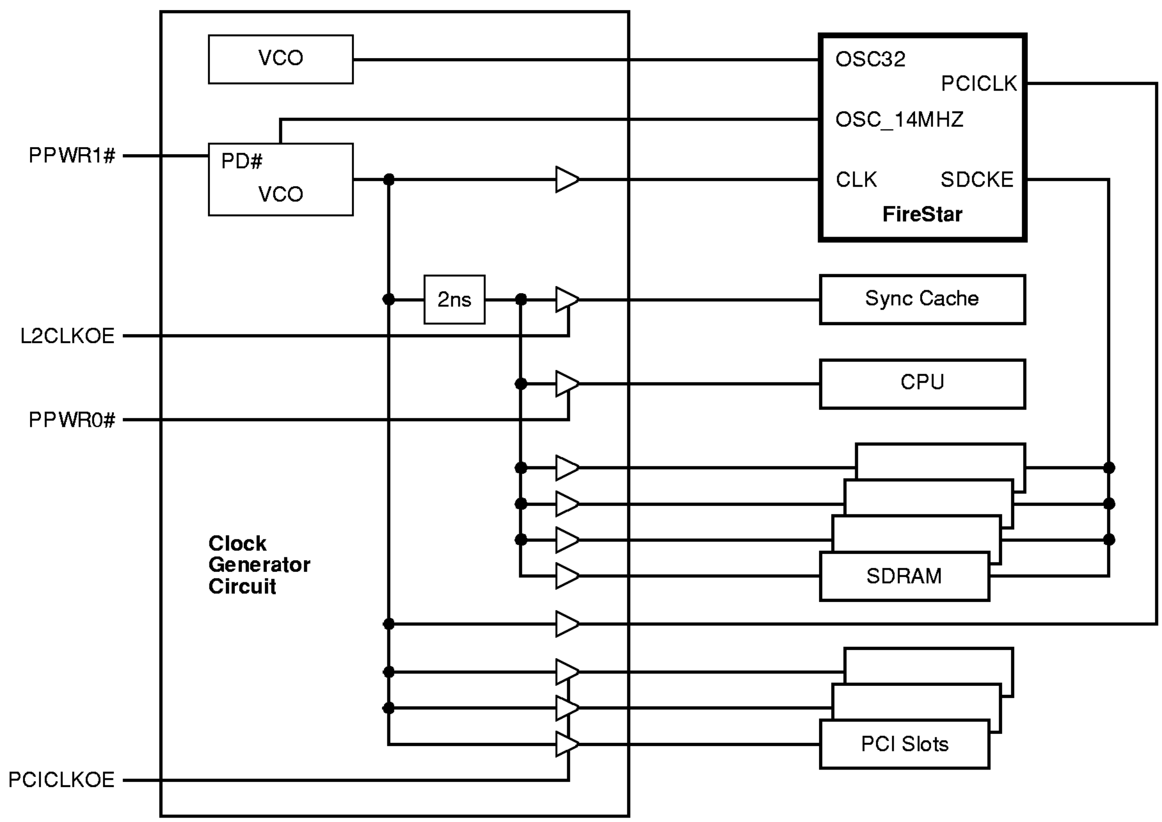
The master CPU clock input to FireStar, CPUCLKIN, is a single phase clock which is used to sample all host CPU synchronous signals and for clocking its internal state machines. CPUCLKIN must be earlier than the clock to the CPU and should lead by 1.5-2.5ns.

Figure 4-2 shows a typical CPU and FireStar clock distribution circuit. PPWR1# is an inverted output of PPWR1 that is output by FireStar that can be used to power off the clock generator in Suspend. Note that in Suspend, all clocks in the system may be powered off except the OSC32 clock input to FireStar. This clock is used by FireStar to track various power management events. PPWR0# can be used to turn off the CPU clock during APM for greater power savings on the CPU in the Stop Clock state. L2CLKOE can be used to dynamically control the clock to the synchronous cache.

4.4.2 PCI Bus Clocks

FireStar requires PCICLK for the PCI interface. PCICLK can be synchronous or asynchronous. Figure 4-2 also shows a typical clock generation and distribution scheme for PCICLK. In order to support synchronous PCI operation, the leading edge of the PCI clock input to FireStar must not lag or lead the leading edge of the CPU clock from which it is derived by more than 1ns. PCICLKOE can be used to turn off the clocks to all PCI devices except FireStar in APM.

Figure 4-2 Typical CPU and FireStar Clock Distribution



Note: All control inputs are high for operation and low for stop.

4.4.2.1 PCI Clock Generation

FireStar provides CLKRUN#-controlled clocks to multiple PCI devices. The clocks are derived from the PCICLKIN line. This clock is fed directly to the system logic. A partial-clock delay is introduced to correct for the output buffer delay and the circuit board trace delay, and the clocks are fed out to one or more output lines.

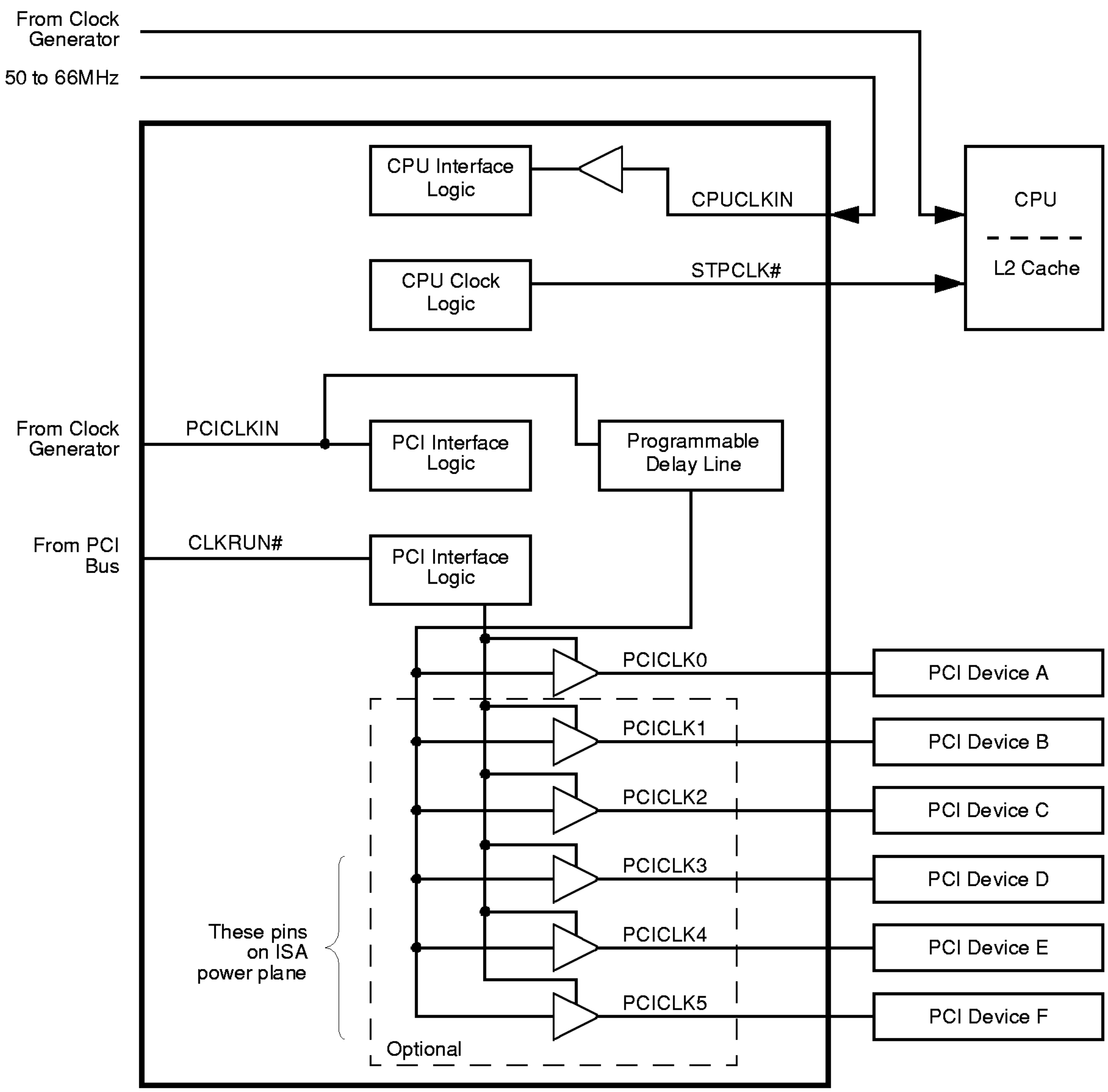
PCICLKIN should be exactly in phase with the CPUCLKIN signal for high-performance synchronous operation. Figure 4-3 illustrates the clocking arrangement.

Buffered PCICLK Outputs

Up to five buffered PCICLK sources are available as strap-selected options on certain output-only pins. For example, if the ATCLK signal is not needed in the system, it can be reassigned at reset as a PCICLK source for one PCI device.

The internal PCI clock skew is corrected through an internal delay line. Programming the delay line introduces any needed delay to account for trace delays to the external devices. This scheme eliminates the need to use external components to adjust clock skew. Note that three of the options are on the ISA power plane, and therefore may not be appropriate if PCI and ISA are at different voltages.

Figure 4-3 System Clock Generation



Each PCICLK output is individually controlled by the CLKRUN# circuitry. In this way, devices that do not support CLKRUN# can be supported along with those that do.

The alternative option to reassigning FireStar pins as PCI clock outputs is to use externally buffered versions of the main PCICLK signal to drive PCI devices.

Programming

The PCICLK outputs 1-5 are enabled as strap-selected features. Once enabled, PCIDV1 68h through 6Bh control the clock delay and stop/start each clock individually (refer to Table 4-3).

On reset, the delay circuit is disabled and the input clock is output directly to the outputs. The internal delay of the chip will cause the output to be delayed from the input by 3-6ns.

Clock Throttling

FireStar fully supports power management by CPU clock throttling. PCI bus clocking is controlled by CLKRUN#, which the chip will attempt to deassert each time the CPU is put into a power-saving mode.

Table 4-3 PCICLK Outputs Programming Registers

7	6	5	4	3	2	1	0
PCIDV1 68h			**PCICLK Control Register 1**				**Default = FFh**
Reserved		PCICLK5: 0 = Disable 1 = Enable	PCICLK4: 0 = Disable 1 = Enable	PCICLK3: 0 = Disable 1 = Enable	PCICLK2: 0 = Disable 1 = Enable	PCICLK1: 0 = Disable 1 = Enable	PCICLK0: 0 = Disable 1 = Enable
PCIDV1 69h			**PCICLK Control Register 2**				**Default = 00h**
Reserved		PCICLK5 affected by CLKRUN#: 0 = No 1 = Yes	PCICLK4 affected by CLKRUN#: 0 = No 1 = Yes	PCICLK3 affected by CLKRUN#: 0 = No 1 = Yes	PCICLK2 affected by CLKRUN#: 0 = No 1 = Yes	PCICLK1 affected by CLKRUN#: 0 = No 1 = Yes	PCICLK0 affected by CLKRUN#: 0 = No 1 = Yes
PCIDV1 6Ah			**PCICLK Skew Adjust Register for PCICLK 0, 1, 2**				**Default = 00h**
Reserved: For PCICLK debug purposes.	Coarse adjustment: 000 = No delay 001 = (PCICLK period ÷2) + ~4ns 010 = (PCICLK period ÷2) + ~8ns 011 = (PCICLK period ÷2) + ~12ns 100 = (PCICLK period ÷2) + ~16ns 101 = (PCICLK period ÷2) + ~20ns 110 = (PCICLK period ÷2) + ~24ns 111 = (PCICLK period ÷2) + ~28ns			Reserved	Fine adjustment: 000 = No delay 001 = Add ~1ns 010 = Add ~2ns 011 = Add ~3ns 100 = Add ~4ns 101 = Add ~5ns 110 = Add ~6ns 111 = Add ~7ns		
Note: If both coarse adjustment and fine adjustment are set to 0 (no delay), PCICLKIN will be routed to PCICLK output with no compensation.							
PCIDV1 6Bh			**PCICLK Skew Adjust Register for PCICLK 3, 4, 5**				**Default = 00h**
Reserved: For PCICLK debug purposes.	Coarse adjustment: 000 = No delay 001 = (PCICLK period ÷2) + ~4ns 010 = (PCICLK period ÷2) + ~8ns 011 = (PCICLK period ÷2) + ~12ns 100 = (PCICLK period ÷2) + ~16ns 101 = (PCICLK period ÷2) + ~20ns 110 = (PCICLK period ÷2) + ~24ns 111 = (PCICLK period ÷2) + ~28ns			Reserved	Fine adjustment: 000 = No delay 001 = Add ~1ns 010 = Add ~2ns 011 = Add ~3ns 100 = Add ~4ns 101 = Add ~5ns 110 = Add ~6ns 111 = Add ~7ns		
Note: If both coarse adjustment and fine adjustment are set to 0 (no delay), PCICLKIN will be routed to PCICLK output with no compensation.							

4.4.3 ISA Bus Clocks

FireStar generates the ISA bus clock (ATCLK) from an internal division of PCICLK. The ATCLK frequency is programmable and can be set to any of the four clock division options through PCIDV1 47h[5:4] (see Table 4-4). This allows the system designer to tailor the ISA bus clock frequency to support a wide range of system designs and performance platforms.

4.4.4 Clock Control

In addition to the PPWR0 clock control, the FireStar power management unit (PMU) provides the L2CLKOE signal as an option on PIO pins. The two signals function in a slightly different manner.

- PPWR0 automatically goes low to turn off the clock to the CPU during APM Doze mode. The toggle occurs only after the PMU has received the Stop Grant cycle from the CPU. Once an interrupt event comes in to wake the system out of Doze mode, PPWR0 goes high again to restart the clock generator. The STPCLK# signal to the CPU remains asserted for an additional 1ms to allow the PLL of the CPU to stabilize.
- L2CLKOE automatically goes low to turn off the clock to the L2 cache any time Stop Grant mode is active on the CPU. As with PPWR0, the toggle occurs only after the PMU has received the Stop Grant cycle from the CPU. However, L2CLKOE stays low until STPCLK# is removed. Therefore, it can be used to save power during CPU clock throttling, when the STPCLK# pin to the CPU is asserted on a periodic basis but the CPU clock is not stopped. PPWR0 does not toggle during clock throttling.

Either one or both of these signals can be used to minimize power consumption in a low-power design. PPWR1 is also available as the master control for all system clocks. It is used to shut off the clock generator completely during Suspend.

4.4.5 L2 Cache Clock Control

FireStar generates the L2CLKOE control to the clock generator to allow it to stop the clock to the L2 cache. This pin is deasserted when the cache is not being accessed, when the CPU is in Stop Grant state. The L2 cache clock runs at all other times, since the cache must be able to latch an upcoming CPU address even before any CPU command lines go active.

L2CLKOE is a programmable pin option and can be assigned to any available PIO pin.

Table 4-4 ATCLK Frequency Control Register Bits

7	6	5	4	3	2	1	0
PCIDV1 47h			**PCI Control Register B - Byte 1**				**Default = 00h**
		ATCLK frequency: 00 = PCICLK ÷4 01 = PCICLK ÷3 10 = PCICLK ÷2 11 = PCICLK					

4.5 Cache Subsystem

The integrated cache controller, which uses a direct-mapped, scheme dramatically boosts the overall performance of the local memory subsystem by caching writes as well as reads (writeback mode). The cache controller also supports 256KB, 512KB, 1MB, and 2MB of synchronous SRAM in a single/double bank configuration. Two programmable non-cacheable regions are provided. The cache controller operates in a non-pipelined or a pipelined mode, with a fixed 32-byte line size (optimized to match a CPU burst linefill) in order to simplify the motherboard design without increasing cost or degrading system performance. The secondary cache operates independently and in addition to the CPU's internal cache.

The cache controller of FireStar has a built-in tag comparator which improves system performance while reducing component count on the system board. The controller features a 64-bit wide data bus with 32-byte CPU burst support. The cache controller supports writeback, adaptive writeback, and write-through schemes.

The cache controller uses a 32-byte secondary cache line size. It supports 3-1-1-1 burst read/write for pipelined synchronous SRAMs. 2-1-1-1 burst read/write cycles are supported for standard synchronous SRAMs at 50MHz. In this case, the ADSC# output of the processor needs to be connected to the ADSC# input of the synchronous SRAM. The 8-bit tag has a “dirty” bit option for the writeback cache.

4.5.1 CPU Burst Mode Control

FireStar fully supports the 64-bit wide data path for the CPU burst read and burst write cycles. The cache and DRAM controllers in FireStar ensure that data is burst into the CPU whenever the CPU requests a burst linefill or a burst write to the system memory.

FireStar contains separate burst counters to support DRAM and external cache burst cycles. The DRAM controller performs a burst for the L2 cache read miss linefill cycle (DRAM to L2 cache and CPU) and the cache controller burst supports the CPU burst linefill (3.3V Pentium and K5 burst linefill and the Cyrix M1 linear burst linefill) for the L2 cache hit cycle (L2 cache to the 3.3V Pentium CPU). Depending on the kind of processor being used, either the 3.3V Pentium quad-word burst address sequencing or the Cyrix M1 quad-word linear burst address sequencing is used for all system memory burst cycles.

4.5.1.1 Cyrix Linear Burst Mode Support

FireStar supports the Cyrix linear burst mode. SYSCFG 17h[0] determines which burst mode is to be implemented, the Intel 3.3V Pentium CPU burst mode or the Cyrix linear burst mode. No additional hardware is required for supporting either of these modes.

When using a synchronous SRAM solution, care must taken that the synchronous SRAM burst protocol complements the processor’s burst protocol.

Table 4-5 shows the burst mode sequence for both of these processors and Table 4-6 highlights the register bits that need to be programmed upon system burst mode selection.

Table 4-5 Burst Modes

1st Address	2nd Address	3rd Address	4th Address
Cyrix Linear Burst Mode			
0	8	10	18
8	10	18	0
10	18	0	8
18	0	8	10
Intel/AMD Burst Mode			
0	8	10	18
8	0	18	10
10	18	0	8
18	10	8	0

Table 4-6 Burst Mode Control Register Bit

7	6	5	4	3	2	1	0
SYSCFG 17h			**PCI Cycle Control Register 2**				**Default = 00h**
							Burst type: 0 = Intel burst protocol 1 = Cyrix linear burst protocol
SYSCFG 0Dh			**Clock Control Register**				**Default = 00h**
					Add one more wait state during PCI master cycle with Intel-type address toggling(1): 0 = No 1 = Yes		

(1) If the PCI master does its address toggling in the style of the Intel 486 burst, rather than a linear burst mode style, then one wait state needs to be added.

4.5.2 Cache Cycle Types

Some cache terminology and cycle definitions that are chipset specific:

The cache hit/miss status is generated by comparing the high-order address bits (for the memory cycle in progress) with the stored tag bits from previous cache entries. When a match is detected and the location is cacheable, a cache hit cycle takes place. If the comparator does not detect a match or a non-cacheable location is accessed (based on the internal non-cacheable region registers), then the current cycle is a cache miss.

A cache hit/miss decision is always made at the end of the first T2 for a non-pipeline cycle and at the end of the first T2P for a pipeline cycle, so the SRAM read/write cycle will begin after the first T2 or T2P. The cacheable decision is based on the DRAM bank decodes and the chipset's configuration registers for non-system memory areas and non-cacheable area definitions. If the access falls outside the system memory area, it is always non-cacheable.

The dirty bit is a mechanism for monitoring coherency between the cache and system memory. Each tag entry has a corresponding dirty bit to indicate whether the data in the represented cache line has been modified since it was loaded from system memory. This allows FireStar to determine whether the data in the system memory is "stale" and needs to be updated before a new memory location is allowed to overwrite the currently indexed cache entry. FireStar supports several Tag/Dirty schemes and those are described in Section 4.5.3.7, Tag and Dirty RAM Implementations, on page 56.

A linefill cycle occurs for a cache read miss cycle. It is a data read of the new address location from the system memory and a corresponding write to the cache. The tag data will also be updated with the new address.

A castout cycle occurs for a cache read miss cycle, but only if the cache line that is being replaced is "dirty". In this cycle, the dirty cache line is read from the cache and written to the system memory. The upper address bits for this cycle are provided by the tag data bits.

A writeback cycle consists of performing a castout cycle followed by a linefill cycle. The writeback cycle causes an entire cache line (32 bytes) to be written back to memory followed by a line burst from the new memory location into the cache and to the CPU simultaneously. The advantages of performing fast write cycles to the cache (for a write hit) typically outweigh the cycle overhead incurred by the writeback scheme.

4.5.3 Cache Operation

The following section describes the cache operation on FireStar.

4.5.3.1 L2 Cache Read Hit

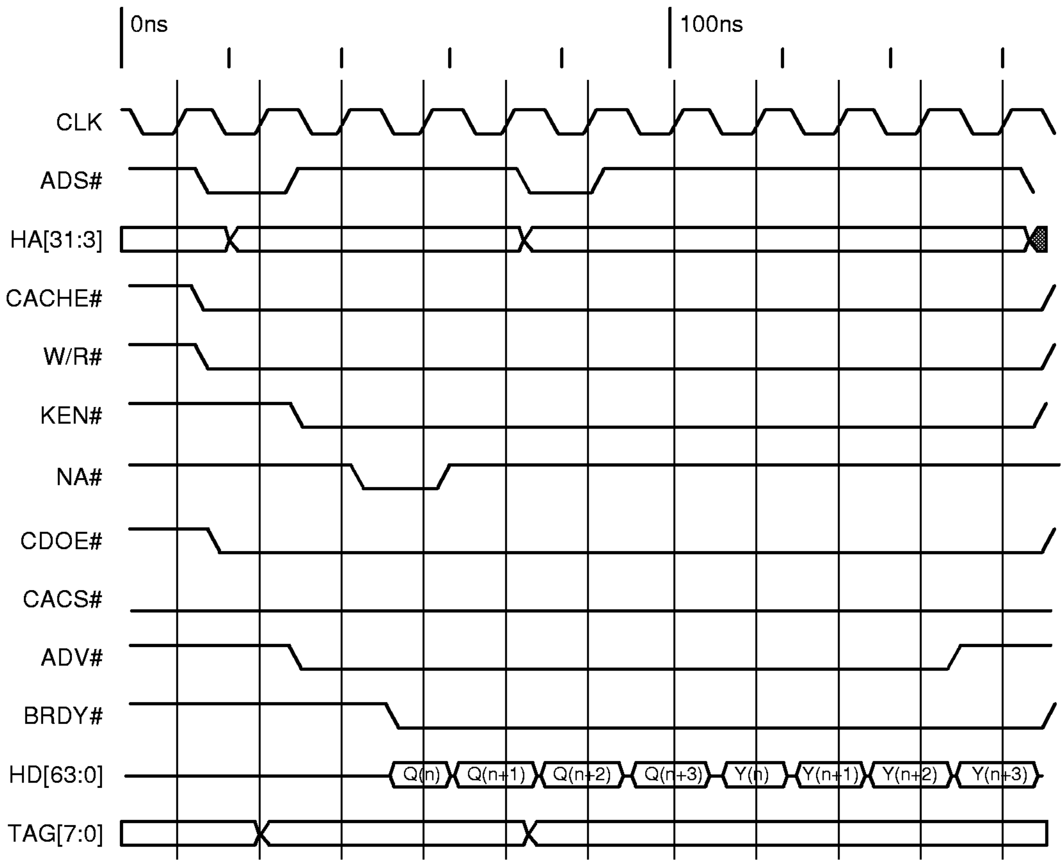
On an L1 read miss and an L2 read hit, the secondary cache provides data to the CPU. FireStar follows the CPU's burst protocol mode (i.e., either linear or non-linear) to fill the processor's internal cache line.

The cache controller will sample CACHE# from the CPU at the end of T1 and perform a burst read if CACHE# is sampled active. The first cache read hit for a cycle is always one wait state. If a read cycle can be converted to a burst, the read cycle is extended for the additional three words continuing at one wait state per cycle. To achieve the burst at this rate, the hit or miss decision must be made before BRDY# is returned to the CPU at the end of the second T2. The cache hit comparator in FireStar compares the data from the tag RAM with the higher address bits from the CPU bus. The output of this comparator generates the BRDY# signal to the 3.3V Pentium CPU. The tag comparator's output is sampled at the end of the first T2, and BRDY# is generated one clock later for cache hits, resulting in a leadoff of three cycles. BRDY# will go inactive to add wait states depending on the wait states programmed. Refer to Table 4-8 for the tag compare table.

The data output for the SRAM is controlled by a separate output enable (CDOE#). The CDOE# generation for the leadoff cycle is based on address bit A3 from the CPU. The output enable CDOE# will be active for the complete cycle.

Figure 4-4 shows an example of an L2 cache read hit cycle using synchronous SRAMs.

Figure 4-4 L2 Cache Read Hit Cycle - Sync SRAMs



4.5.3.2 L2 Cache Write Hit Cycle

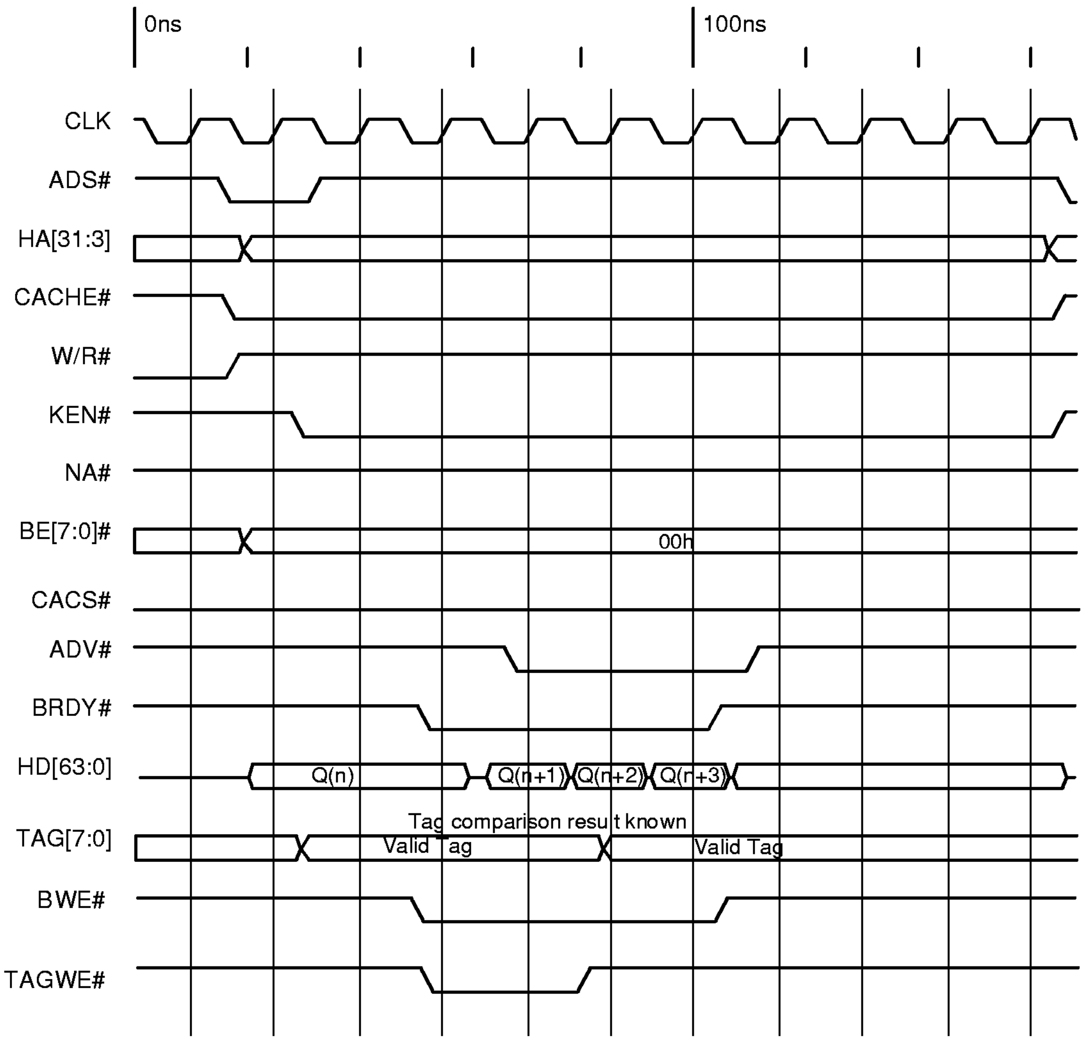
Write-through Mode: In this mode, data is always written to the L2 cache and to the system memory. The dirty bit is not used. When the write to the system memory is completed, BRDY# is returned to the CPU.

Writeback Mode: For a write hit case, the data is written only to the L2 cache (the system memory is not updated) and the dirty bit is always made dirty. The cache controller will sample CACHE# from the CPU at the end of T1 and execute a burst write if CACHE# is sampled active, otherwise the cycle will end in a single write. In this mode, the write cycle is completed in a 3-1-1-1 burst.

For writes, only the byte requested by the CPU can be written to the cache. This is done by using the BEx# from the CPU to control the SRAM write enable signals.

Figure 4-5 shows an example of a write hit burst cycle in writeback mode using synchronous SRAMs.

Figure 4-5 Write Hit Burst Cycle for Writeback Mode - Sync SRAM



4.5.3.3 L2 Cache Read Miss

Writeback Mode: There are two cache read miss cases depending on the status of the dirty bit.

CASE 1: Read miss of a “clean” cache line.

In this case, only a linefill cycle is executed. The L2 cache line that is to be replaced with a new line from the DRAM will just be overwritten. The linefill cycle is done by reading the new data from the system memory first and then the data is simultaneously written to both the CPU and the secondary cache.

The sequence for CASE 1 linefill is: System memory read ⇨ write to the L2 cache + CPU read.

The cache controller will update the tag data bits and the dirty bit in the background during the linefill cycle. At the end of T1, if the CACHE# signal from the CPU is negated, a linefill cycle will not be executed. Instead, only the eight bytes requested by the CPU will be read from the system memory. The tag and the dirty bit will not be updated.

CASE 2: Read miss with cache line dirty.

The cache line for this case has been modified and only the L1 and L2 cache have the updated copy of the data. Before this line is overwritten in the cache, the modified line must first be written to the system memory by performing a castout cycle. After the completion of the castout cycle, a linefill cycle is executed. The linefill cycle is performed by reading the new data from the system memory and then simultaneously writing this data to the CPU and the secondary cache.

The sequence for CASE 2 is: Read the dirty line from L2 cache ⇨ write to the system memory ⇨ new line read from system memory ⇨ write to the L2 cache + CPU read.

The cache controller will update the tag data bits and the dirty bit in the background during the castout cycle. If the CACHE# signal from the CPU is inactive, then the eight bytes requested by the CPU will be read from the system memory. The tag and the dirty bit are not updated.

4.5.3.4 L2 Cache Write Miss

Writeback or Write-through Cases: The data is not written to the SRAM and the tag data remains unchanged. The data is written only to the system memory.

If the write buffer and DRAM posted write is enabled then is available, it is stored there and the cycles are posted writes to the DRAM. If the target is on the PCI or ISA bus, the cache controller will not be active.

4.5.3.5 Write Policies

Any of the following three write policies supported by FireStar can be chosen: writeback, write-through, and adaptive writeback, by programming SYSCFG 02h[5:4] and SYSCFG 08h[1] (as shown in Table 4-7).

Depending on the state of these bits and the type of DRAM cycle that would be required to complete the write hit cycle, the cache controller decides whether to update the DRAM memory, however, the cache is always updated. The adaptive writeback policy tries to reduce the disadvantages of both the write-through and the writeback schemes to a minimum. The adaptive writeback scheme converts a write hit cycle to a write through cycle only if the address location being written to corresponds to a page hit. In this manner, this scheme incurs a four CLK penalty for a burst write cycle but it saves a 13 CLK penalty (for a castout cycle) that would have occurred later due to a read miss access. There are two adaptive writeback modes.

Write-Through on Page Hit and RAS# Active (AWB Mode 1)

In this mode, the data is written through to the DRAM on a write hit if the address being written to causes a page hit and the corresponding RAS# signal is active. The data will not be written through if, either the RAS# is inactive or if it is a page miss. In this case, the write hit cycle completes in the same manner as in a writeback scheme.

Write-Through on Page Hit (AWB Mode 2)

In this mode, data is written through to the DRAM on a write hit if the address being written to causes a page hit. RAS# being active/inactive does not come into consideration when making this decision.

Table 4-7 Register Bits Associated with Write Policies

7	6	5	4	3	2	1	0
SYSCFG 02h			**Cache Control Register 1**				**Default = 00h**
		L2 cache write policy: 00 = L2 cache write-through 01 = Adaptive writeback Mode 1 10 = Adaptive writeback Mode 2 11 = L2 cache writeback					

Table 4-7 Register Bits Associated with Write Policies (cont.)

7	6	5	4	3	2	1	0
SYSCFG 08h			**CPU Cache Control Register**				**Default = 00h**
							BIOS area cacheability in L1 cache: Determines if system BIOS area E0000h-FFFFFh (if SYSCFG 04h[2] = 1) or F0000h-FFFFFh (if SYSCFG 04h[2] = 0), and video BIOS area C0000h-C7FFFh is cacheable in L1 or not. 0 = Cacheable 1 = Not Cacheable
SYSCFG 04h			**Shadow RAM Control Register 1**				**Default = 00h**
					E0000h-EFFFFh range selection: Determines whether this region will be treated like the F0000 BIOS area or whether it will always be non-cacheable. 0 = E0000h-EFFFFh area will always be non-cache-able 1 = E0000h-EFFFFh area will be treated like the F0000h BIOS area. If this bit is set, then SYSCFG 06h[3:2] and [1:0] Should be set identically.		

Table 4-7 Register Bits Associated with Write Policies (cont.)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">SYSCFG 06h</td><td colspan="4">Shadow RAM Control Register 3</td><td colspan="2">Default = 00h</td></tr>
<tr><td></td><td></td><td>C0000h-C7FFFh cacheability:
0 = Not cacheable
1 = Cacheable in L1 and L2 (L1 disabled by SYSCFG 08h[0])</td><td>F0000h-FFFFFh cacheability:
0 = Not cacheable
1 = Cacheable in L1 and L2 (L1 disabled by SYSCFG 08h[0])</td><td colspan="2">F0000h-FFFFFh read/write control:
00 = Read/write PCI bus
01 = Read from DRAM/write to PCI
10 = Read from PCI/write to DRAM
11 = Read/write DRAM
If SYSCFG 04h[2] = 1, then the E0000h-EFFFFh read/write control should have the same setting as this.</td><td colspan="2">E0000h-EFFFFh read/write control:
00 = Read/write PCI bus
01 = Read from DRAM/write to PCI
10 = Read from PCI/write to DRAM
11 = Read/write DRAM</td></tr>
</table>

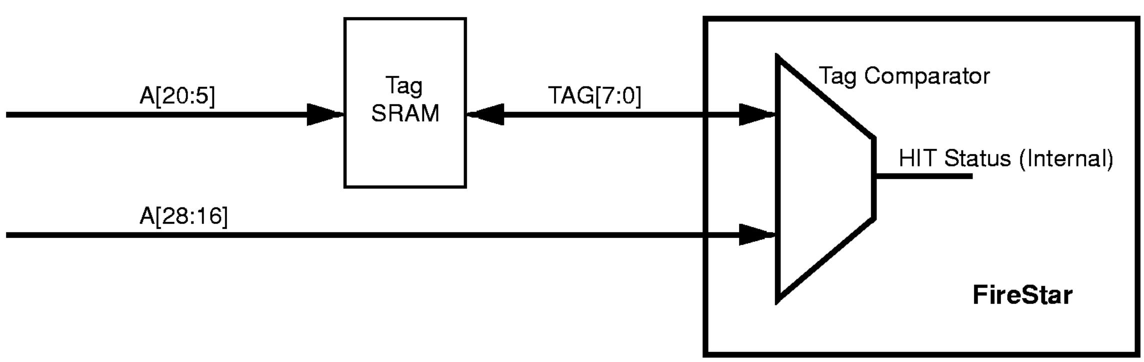
4.5.3.6 Tag Compare Table

The upper address bits used to compare for a L2 cache hit status will depend on the total L2 cache size. Table 4-8 shows the address bits from the CPU bus and the tag data bit used in the tag comparator of FireStar. Figure 4-6 shows the block diagram of the L2 cache tag structure.

Table 4-8 Tag Compare Table

Tag Data	L2 Cache Size					
	64KB	**128KB**	**256KB**	**512KB**	**1MB**	**2MB**
TAG0	A16	A24	A24	A24	A24	A24
TAG1	A17	A17	A25	A25	A25	A25
TAG2	A18	A18	A18	A26	A26	A26
TAG3	A19	A19	A19	A19	A27	A27
TAG4	A20	A20	A20	A20	A20	A28
TAG5	A21	A21	A21	A21	A21	A21
TAG6	A22	A22	A22	A22	A22	A22
TAG7	A23	A23	A23	A23	A23	A23
Dirty Bit	Dirty	Dirty	Dirty	Dirty	Dirty	Dirty

Figure 4-6 Internal Tag Comparator Block Diagram



4.5.3.7 Tag and Dirty RAM Implementations

There are various tag/dirty RAM implementations supported by FireStar. Table 4-9 shows the tag and dirty RAM register programmable bits located in SYSCFG 16h.

Combined Tag/Dirty RAM Implementation

There are various ways of achieving a combined tag/dirty RAM implementation.

A 32Kx9 SRAM can be used to implement eight tag bits and one dirty bit. In this case, the TAGWE# signal from FireStar is used to update both the tag and dirty information. The OE# signal of the 32Kx9 SRAM can be connected to the DIRYTWE# signal from FireStar or it can be tied to GND. The DIRTYI signal FireStar becomes a bidirectional signal and it now serves as the dirty I/O bit. This scheme is shown in Figure 4-7.

A 32Kx8 SRAM can be used, wherein seven bits are used for the tag RAM and one bit is used for the dirty RAM. In this case, the TAGWE# signal from FireStar is used to update both the tag and dirty information. The OE# of the 32Kx8 SRAM can be connected to the DIRYTWE# signal from FireStar or it can be tied to GND. TAG[7:1] convey the tag information and TAG0 becomes the dirty I/O bit. In this scheme, the amount of main memory that can be cached reduces by half as compared to an 8-bit tag implementation. This scheme is shown in Figure 4-8.

A 32Kx8 SRAM can be used to implement the eight tag bits and another 32Kx8 SRAM used to implement the single dirty I/O bit. This scheme is identical to the 32Kx9 implementation and is shown in Figure 4-9.

Table 4-9 Tag/Dirty RAM Control Register Bits

7	6	5	4	3	2	1	0
SYSCFG 16h			**Dirty/Tag RAM Control Register**				**Default = A0h**
This bit along with bit 5 and PCICLK3 strap define DIRTY, CMD# as PCICLK3 on the CMD# pin, and CACS# or DIRTY options on the CACS# pin.(1)		Tag RAM size selection: 0 = 8-bit 1 = 7-bit (Default) Selects CACS# for 7-bit and DIRTY for 8-bit tag(1)	Single write hit leadoff cycle in a combined Dirty/Tag implementation 0 = 5 cycles 1 = 4 cycles	Pre-snoop control: 0 = Pre-snoop for starting address 0 only 1 = Pre-snoop for all addresses except those on the line boundary			

(1) **ROMCS#:KBDCS# strapped for CMD#:**

Bits 7 & 5	CACS# Pin	CMD# Pin
00	CACS#	DIRTY
01	CACS#	DIRTY
10	DIRTY	CMD#
11	CACS#	CMD#

(1) **ROMCS#:KBDCS# strapped for PCICLK3:**

Bits 7 & 5	CACS# Pin	CMD# Pin
00	DIRTY	PCICLK3
01	CACS#	PCICLK3
10	DIRTY	PCICLK3
11	CACS#	PCICLK3

Figure 4-7 32Kx9 Combined Tag/Dirty RAM Implementation

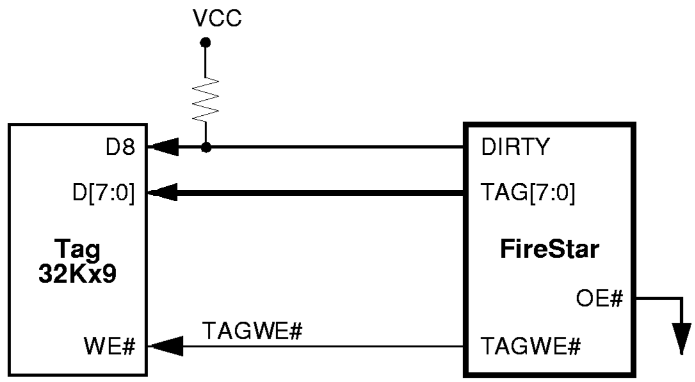


Figure 4-8 32Kx8 Combined Tag/Dirty RAM Implementation

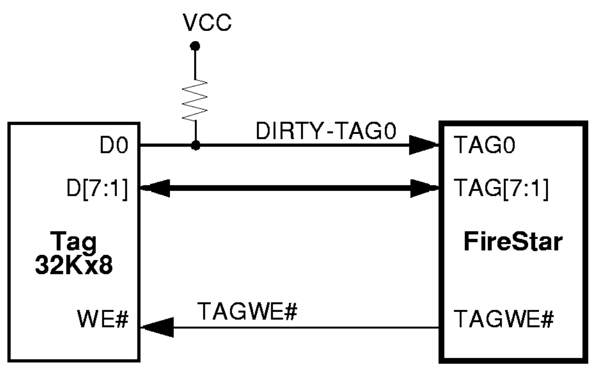
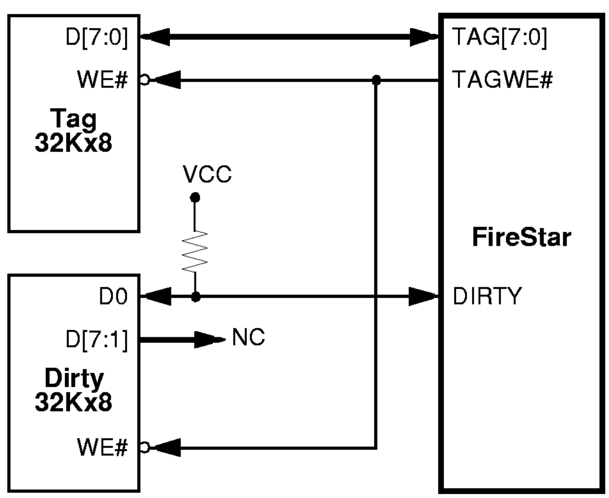


Figure 4-9 32Kx8 and 32Kx8 Combined Tag/Dirty RAM Implementation (Separate Devices)



4.5.3.8 Cache Initialization

On power-up, the tag RAM will contain random data and the L2 cache will contain no valid data. Therefore, the cache must be initialized before it is enabled.

Initializing Procedure 1: The cache is initialized by configuring the cache controller to the write-through mode. This will cause all the cache read miss cycles to fill the cache with valid data. This can be done by reading a block of system memory that is greater than or equal to the size of the cache. Once the cache is initialized, it is always valid. After this is done, the L2 cache can be set up for writeback operation by initializing the dirty bits. This is done by first enabling the cache controller to the writeback mode. Then, by reading a block of system memory that is greater than or equal to *twice* the size of the cache, the dirty bits will be cleared and the L2 cache will be valid.

Initializing Procedure 2: This procedure uses the cache controller in Test Mode 1 and Test Mode 2 as defined in SYSCFG 02h[3:2] and 07h[7:0]. (Refer to Table 4-10.)

The upper bits of an address is written to SYSCFG 07h. The cache controller is now set to Test Mode 2. Writing a block equal to the size of the cache to the system memory will write the contents of SYSCFG 07h to the tag. The cache controller is now configured in the write-through mode and reading a block of system memory equal to the size of the cache will make the data in the cache valid. Next, by reading a block of system memory which is greater than or equal to twice the size of the cache, the dirty bits will be cleared and the L2 cache will be valid.

Disabling the Cache: Disabling of a writeback cache **cannot** be done by just turning off SYSCFG 02h[3:2]. There may still be valid data in the cache that has not been written to the system memory. Disabling writeback cache without flushing this valid data usually causes a system crash.

This situation can be avoided by first reading a cacheable memory block *twice* the size of the cache. "Twice the size" of the cache is required to make sure every location gets a read miss, which will cause a castout cycle if the cache line is dirty. The cache can then be disabled. ***Note: No writes should occur during this process.***

4.5.3.9 Write Back Cache with DMA/ISA Master/PCI Master Operation

The L1 and the L2 cache contain the only valid copy (modified) of the data. FireStar will execute an inquire cycle to the L1 cache for all master accesses to the system memory area. This will increase the bus master cycle time for every access to the system memory which will also decrease the bus master performance. FireStar provides the option of a snoop-line comparator (snoop filtering) to increase the performance of a bus master with the L1 cache.

L1 Cache Inquire Cycle: This cycle begins with the CPU relinquishing the bus with the assertion of BOFF#. On sampling HLDA active, FireStar will assert AHOLD. The address will flow from the master to the CPU bus and FireStar will assert EADS# for one CPUCLK. If the CPU does not respond with the assertion of HITM#, FireStar will complete the cycle from the L2 cache or the system memory. If HITM# was asserted, FireStar will expect a castout cycle from the L1 cache.

Table 4-10 Test Mode Selection/Control Bits

7	6	5	4	3	2	1	0
SYSCFG 02h			**Cache Control Register 1**				**Default = 00h**
				L2 cache operating mode select: 00 = Disable 01 = Test Mode 1; External Tag Write (Tag data write-through SYSCFG 07h) 10 = Test Mode 2; External Tag Read (Tag data read from SYSCFG 07h) 11 = Enable L2 cache			
SYSCFG 07h			**Tag Test Register**				**Default = 00h**
- Data from this register is written to the tag, if in Test Mode 1 (refer to SYSCFG 02h). - Data from the tag is read into this register, if in Test Mode 2 (refer to SYSCFG 02h).							

DMA/Master Read Cycle: Table 4-11 shows the action taken by FireStar based on the L1 and L2 cache status for bus master reads from the system memory area. The L1 cache castout cycle will be completed in the burst order provided by the CPU and will be written to the L2 cache or the system memory based on the L2 cache status. The required bytes are then read back for the completion of the master read cycle. A read hit in the L1 cache will always invalidate the L1 cache line. Refer to Figures 4-10 and 4-11.

DMA/Master Write Cycle: Table 4-12 shows the action taken by FireStar based on the L1 and L2 cache status for bus master writes to the system memory area. A master write to the L2 cache will always be in the write-through mode. The L1 cache castout cycle will be completed in the CPU burst sequence and the data will be written to the L2 cache or to the system memory based on the L2 cache status. Data from the master is always written to the system DRAM memory and is written to the L2 cache only if it is a L2 cache hit. Refer to Figure 4-12.

Table 4-11 DMA/Master Read Cycle Summary

DMA/Master Read Cycle					
L1 Cache	**L2 Cache**	**Data Source**	**Type of Cycle for L1 Cache**	**Type of Cycle for L2 Cache**	**Type of Cycle for DRAM**
Hit	Hit	L2 Cache	Invalidate	Read the Bytes Requested	No Change
hitM	Hit	L1 Cache	Castout, invalidate	Write CPU Data, Read Back the Bytes Requested	No Change
Hit	Miss	DRAM	Invalidate	No Change	Read the Bytes Requested
hitM	Miss	L1 Cache	Castout, invalidate	No Change	Write CPU Data, Read Back the Bytes Requested
Miss	Hit	L2 Cache	No Change	Read the Bytes Requested	No Change
Miss	Miss	DRAM	No Change	No Change	Read

Note: hitM - L1 cache modified

Table 4-12 DMA/Master Write Cycle Summary

DMA/Master Write Cycle					
L1 Cache	**L2 Cache**	**Data Destination**	**Type of Cycle for L1 Cache**	**Type of Cycle for L2 Cache**	**Type of Cycle for DRAM**
Hit	Hit	DRAM, sec	Invalidate	Write Master Data	Write Master Data
hitM	Hit	DRAM, sec	Castout, Invalidate	Write CPU Data, Write Master Data	Write Master Data
Hit	Miss	DRAM	Invalidate	No Change	Write Master Data
hitM	Miss	DRAM	Castout, Invalidate	No Change	Write CPU Data, Write Master Data
Miss	Hit	DRAM, sec	No Change	Write Master Data	Write Master Data
Miss	Miss	DRAM	No Change	No Change	Write Master Data

Figure 4-10 ISA DMA/Master Read (L1 cache with non-modified line)

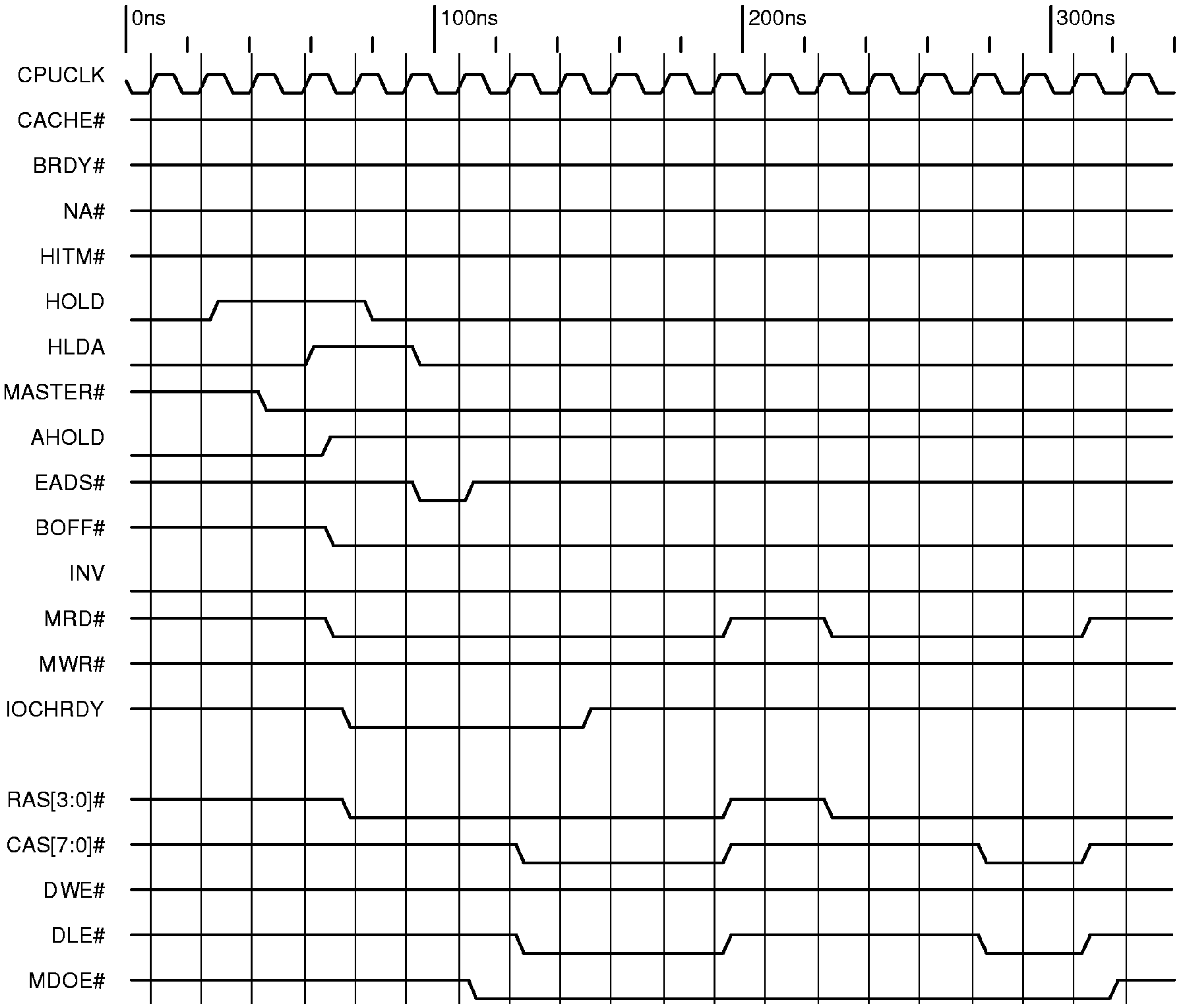


Figure 4-11 ISA DMA/Master Read (L1 cache with modified line)

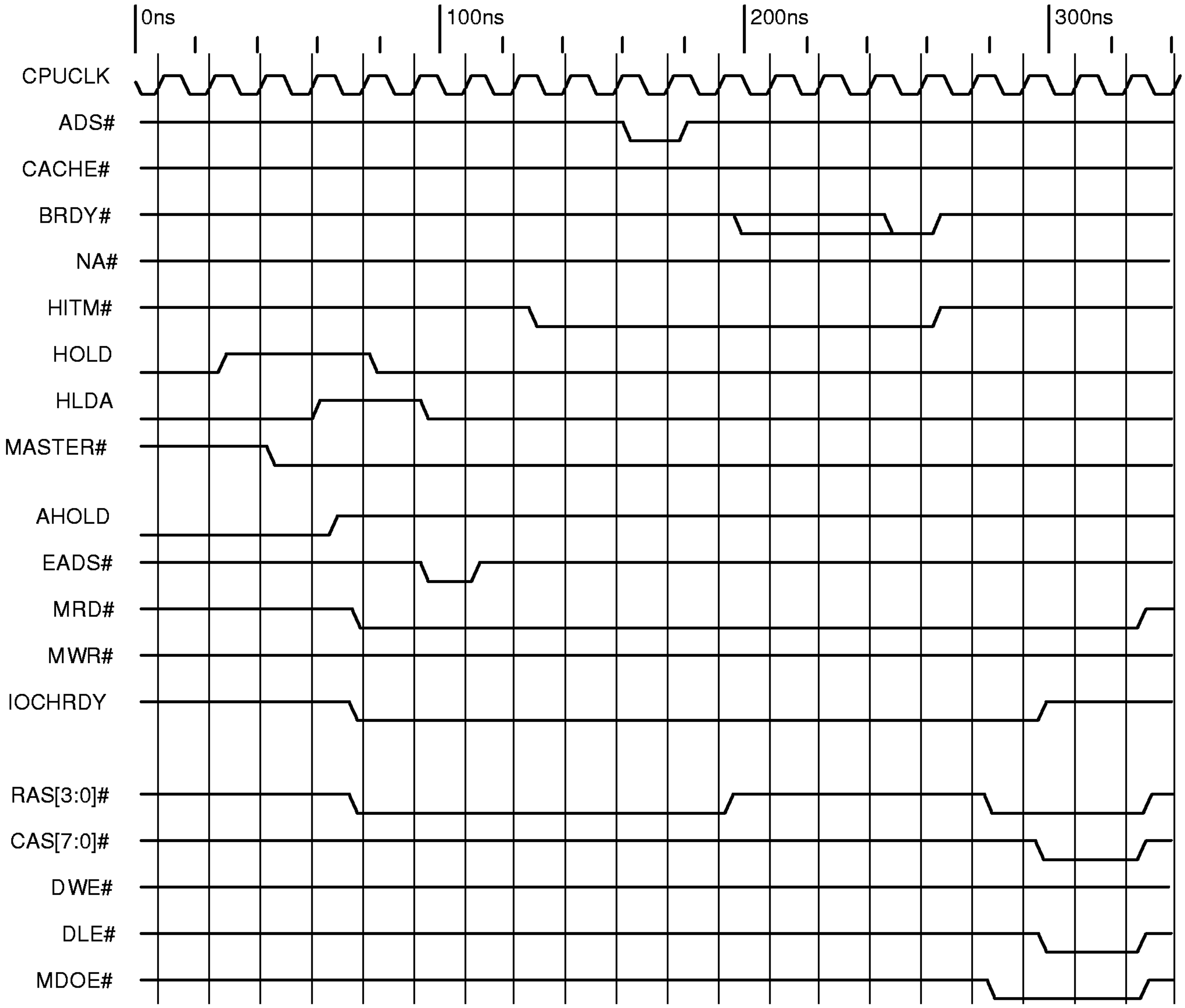
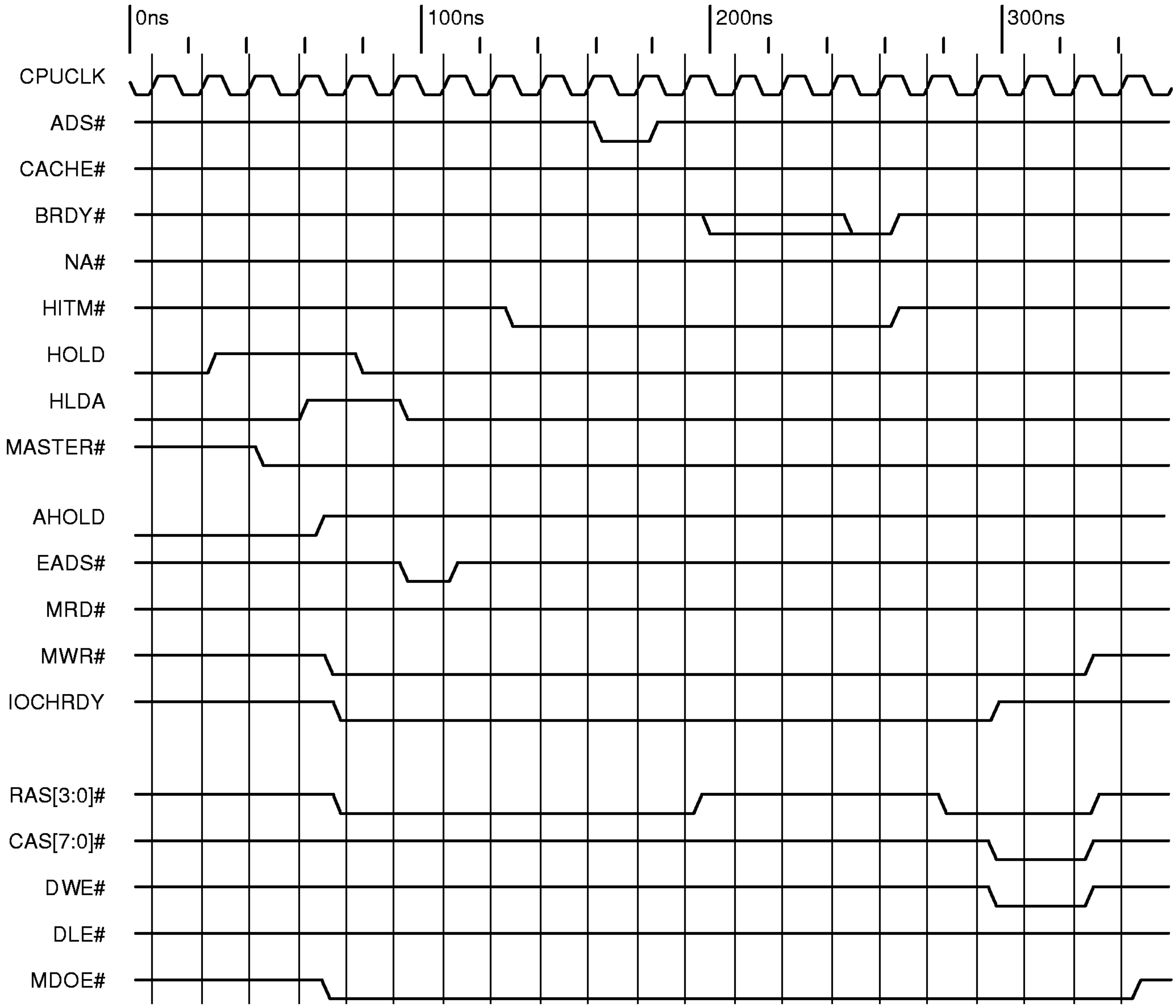


Figure 4-12 ISA DMA/Master Write (L1 cache with modified line)



4.5.4 Shadow ROM and BIOS Cacheability

When using FireStar, the procedures listed below should be followed for proper setup and configuration of shadow RAM utilities.

1. Enable ROMCS# generation for the segment to be shadowed. Although the F0000h-FFFFFh segment defaults to ROMCS# generation, the C, D, and E0000h ROM segments must have ROMCS# generation enabled by setting the appropriate bits in PCIDV1 4Ah and 4Bh.
2. Enable ROM contents to be copied into DRAM. To do this, the appropriate bits in SYSCFG 04h, 05h, and 06h should be set. These bits must be set so that reads from these segments will be executed out of ROM but will be written to DRAM.
3. Enable shadow RAM areas to permit DRAM read/write accesses. At this point, the ROMCS# generation bits that were previously necessary to access the original ROM code, must be disabled.
4. Write protect shadow RAM areas. To do this, the appropriate bits in SYSCFG 04h, 05h, and 06h should be set. These bits must be set so that reads from these segments will be executed out of DRAM, but writes will be directed to the ROM.

5. Cache shadow RAM areas in L2/L1 caches (optional). Caching of the individual code segments can be accomplished by setting the appropriate bits in SYSCFG 06h. Although write protection control for the L2 cache is provided, the L1 cache does not have a write protection mechanism and the ROM code may be overwritten or modified if stored in the L1 cache.

4.5.4.1 Cacheability and Write Protection

FireStar allows certain ROM areas to be cacheable. C0000h-C7FFFh, E0000h-EFFFFh, and F0000h-FFFFFh have separate cache-related controls.

Table 4-13 highlights the appropriate programming register bits.

Table 4-13 Shadow ROM and BIOS Cacheability / Write Protection Control Bits

7	6	5	4	3	2	1	0
PCIDV1 4Ah			**ROM Chip Select Register 1**				**Default = 00h**
ROMCS# for F8000h-FFFFFh: 0 = Enable 1 = Disable	ROMCS# for F0000h-F7FFFh: 0 = Enable 1 = Disable	ROMCS# for E8000h-EFFFFh: 0 = Disable 1 = Enable	ROMCS# for E0000h-E7FFFh 0 = Disable 1 = Enable	ROMCS# for D8000h-DFFFFh: 0 = Disable 1 = Enable	ROMCS# for D0000h-D7FFFh: 0 = Disable 1 = Enable	ROMCS# for C8000h-CFFFFh: 0 = Disable 1 = Enable	ROMCS# for C0000h-C7FFFh: 0 = Disable 1 = Enable
PCIDV1 4Bh			**ROM Chip Select Register 2**				**Default = 00h**
ROMCS# for FFFF8000h-FFFFFFFFh: 0 = Enable 1 = Disable	ROMCS# for FFFF0000h-FFFF7FFFh: 0 = Enable 1 = Disable	ROMCS# for FFFE8000h-FFFEFFFFh: 0 = Disable 1 = Enable	ROMCS# for FFFE0000h-FFFE7FFFh: 0 = Disable 1 = Enable	ROMCS# for FFFD8000h-FFFDFFFFh: 0 = Disable 1 = Enable	ROMCS# for FFFD0000h-FFFFD7FFFh: 0 = Disable 1 = Enable	ROMCS# for FFFC8000h-FFFCFFFFh: 0 = Disable 1 = Enable	ROMCS# for FFFC0000h-FFFC7FFFh: 0 = Disable 1 = Enable
SYSCFG 04h			**Shadow RAM Control Register 1**				**Default = 00h**
CC000h-CFFFFh read/write control: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM		C8000h-CBFFFh read/write control: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM			E0000h-EFFFFh range selection: Determines whether this region will be treated like the F0000 BIOS area or whether it will always be non-cacheable. 0 = E0000h-EFFFFh area will always be non-cacheable 1 = E0000h-EFFFFh area will be treated like the F0000h BIOS area. If this bit is set, then SYSCFG 06h[3:2] and [1:0] Should be set identically.	C0000h-C7FFFh read/write control: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM	

Table 4-13 Shadow ROM and BIOS Cacheability / Write Protection Control Bits (cont.)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">SYSCFG 05h</td><td colspan="4">Shadow RAM Control Register 2</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2">DC000h-DFFFFh read/write control:
00 = Read/write PCI bus
01 = Read from DRAM/write to PCI
10 = Read from PCI/write to DRAM
11 = Read/write DRAM</td><td colspan="2">D8000h-DBFFFh read/write control:
00 = Read/write PCI bus
01 = Read from DRAM/write to PCI
10 = Read from PCI/write to DRAM
11 = Read/write DRAM</td><td colspan="2">D4000h-D7FFFh read/write control:
00 = Read/write PCI bus
01 = Read from DRAM/write to PCI
10 = Read from PCI/write to DRAM
11 = Read/write DRAM</td><td colspan="2">D0000h-D3FFFh read/write control:
00 = Read/write PCI bus
01 = Read from DRAM/write to PCI
10 = Read from PCI/write to DRAM
11 = Read/write DRAM</td></tr>
<tr><td colspan="2">SYSCFG 06h</td><td colspan="4">Shadow RAM Control Register 3</td><td colspan="2">Default = 00h</td></tr>
<tr><td></td><td></td><td>C0000h-C7FFFh cacheability:
0 = Not cacheable
1 = Cacheable in L1 and L2 (L1 disabled by SYSCFG 08h[0])</td><td>F0000h-FFFFFh cacheability:
0 = Not cacheable
1 = Cacheable in L1 and L2 (L1 disabled by SYSCFG 08h[0])</td><td colspan="2">F0000h-FFFFFh read/write control:
00 = Read/write PCI bus
01 = Read from DRAM/write to PCI
10 = Read from PCI/write to DRAM
11 = Read/write DRAM
If SYSCFG 04h[2] = 1, then the E0000h-EFFFFh read/write control should have the same setting as this.</td><td colspan="2">E0000h-EFFFFh read/write control:
00 = Read/write PCI bus
01 = Read from DRAM/write to PCI
10 = Read from PCI/write to DRAM
11 = Read/write DRAM</td></tr>
<tr><td colspan="2">SYSCFG 0Eh</td><td colspan="4">PCI Master Burst Control Register 1</td><td colspan="2">Default = 00h</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td>Write protection for L1 BIOS:
0 = No
1 = Yes</td><td></td></tr>
</table>

Both system DRAM and shadow RAM are cacheable in both the primary (L1) and/or secondary (L2) cache. Of these two areas, only the shadow RAM areas (system BIOS, video BIOS and DRAM) have the capability of being write-protected (Non-shadowed BIOS ROM areas are implicitly write-protected). Since the possibility exists that write-protected shadow RAM can be cached, there also exists the possibility that this data might be modified inside the cache and subsequently executed. To prevent this from occurring, an explicit control mechanism must be used that prevents the unexpected from happening. There are three methods for controlling write protection in FireStar. These methods are discussed next and summarized in Table 4-14.

METHOD 1: In this method, the write protected areas are **not** cached in the L1 or the L2 cache. This is implemented by driving KEN# high for the first word with BRDY#, which will cause the CPU to not cache the data in its L1 cache and not do burst cycles. Data in the L2 cache is also not updated, so all reads and writes to this area will go directly to or from the system memory or to/from system BIOS/video BIOS (if they are not shadowed).

METHOD 2: In this method, the write protected areas can be cached in the L2 cache but not in the L1 cache. This is implemented by driving KEN# high for the first word with BRDY#, which will cause the CPU to not cache the data in the L1 cache or do a burst cycle. This data can then be stored in the L2 cache, but only subsequent read requests by the CPU are serviced (discarding all writes), thus effectively write-protecting the data in the L2 cache. Read miss cycles are serviced by first performing a linefill burst from the DRAM into the L2 cache and then performing a normal non-cacheable (and non-burst) cycle to the CPU. In this method, writes to the system memory and to the L2 cache are write protected.

METHOD 3: This method is implemented by driving EADS# high during the read cycle. Data read from write protected areas are stored in both the L1 and L2 caches. Accesses from the CPU that are L2 cache read hits are serviced in burst mode and L2 cache read miss cycles are serviced by first performing a linefill burst read to the L2 cache from the write protected area and then performing a normal burst cycle to the CPU. Write cycles from the CPU to these areas are write-through and are discarded by the cache controller of the 82C700. ***However, L1 cache writes occur internally to the CPU in this mode and are therefore not write protected***. Table 4-15 shows the register bit (SYSCFG 08h[0]) associated with this function.

Table 4-14 Cacheability Methods

	System DRAM		System BIOS		Video BIOS		Write Enabled Shadow RAM		Write Protected Shadow RAM	
Method	**Read**	**Write**	**Read**	**Write**	**Read**	**Write**	**Read**	**Write**	**Read**	**Write**
1	L1,L2	L1,L2	Single	None	Single	None	L1,L2	L1,L2	Single	None
2	L1,L2	L1,L2	L2	None	L2	None	L1,L2	L1,L2	L2	None
3	L1,L2	L1,L2	L1,L2	L1	L1,L2	L1	L1,L2	L1,L2	L1,L2	L1

Note: L1 = accessible to primary cache
L2 = accessible to secondary cache
None = no cycle performed (or discard)
Single = single word (non-burst) cycle

Table 4-15 SYSCFG 08h[0]

7	6	5	4	3	2	1	0
SYSCFG 08h			**CPU Cache Control Register**				**Default = 00h**
							BIOS area cacheability in L1 cache: Determines if system BIOS area E0000h-FFFFFh (if SYSCFG 04h[2] = 1) or F0000h-FFFFFh (if SYSCFG 04h[2] = 0), and video BIOS area C0000h-C7FFFh is cacheable in L1 or not. 0 = Cacheable 1 = Not Cacheable

4.5.4.2 Remapping of Reset Vector to Shadow DRAM

FireStar allows the reset instruction segment to point to DRAM instead of ROM if desired. This feature allows the soft reset generation of address FFFFFFF0h to point to BIOS shadowed DRAM instead of BIOS in ROM. This feature is enabled through PCIDV1 4Fh[0] as shown in Table 4-16.

Table 4-16 Reset Vector to Shadow DRAM Register Bit

7	6	5	4	3	2	1	0
PCIDV1 4Fh			**Miscellaneous Control Register C - Byte 1**				**Default = 20h**
							BIOS access after soft reset: 0 = ROM 1 = DRAM

4.5.5 SRAM Support

FireStar supports many varieties of synchronous SRAMs. Table 4-17 shows the signals associated with L2 cache SRAM.

In addition to the standard synchronous SRAMs, FireStar supports pipelined synchronous SRAMs as well as the Sony Sonic-2WP (cache module). Table 4-23 at the end of this section gives an SRAM cycle comparison chart.

Table 4-17 L2 Cache Signal Set

Signal Name	Signal Description
TAG[7:0]	**Tag Data Lines:** TAG0 becomes the dirty bit when CACS# is used.
TAGWE#	**Tag Write Enable Line**
CACS#+DIRTY	**Cache Chip Select or Dirty**: Using CACS# save power, but separate DIRTY allows 8-bit tag to expand cacheable area.
GWE#	**Global Write Enable**: Write command to L2 cache indicating that all bytes will be written.
BWE#	**Byte Write Enable:** Write command to L2 cache indicating that only bytes selected by BE[7:0]# will be written.
ADSC#	**Controller Address Strobe**: For a synchronous L2 cache operation, this pin is connected to the ADSC# input of the synchronous SRAMs.
ADV#	**Advance Output**: For synchronous cache L2 operation, this pin becomes the advance output and is connected to the ADV# input of the synchronous SRAMs.
CDOE#	**Cache Output Enable**: This signal is connected to the output enables of the SRAMs of the L2 cache in both banks to enable data read.

4.5.5.1 SRAM Requirements

Table 4-18 gives configuration parameters, while Tables 4-19 and 4-20 outline the read/write cycle lengths and their speed requirements.

4.5.5.2 Pipelined Synchronous SRAM Support

Pipelined synchronous SRAMs are less expensive than their counterpart BiCMOS synchronous SRAMs (standard synchronous SRAMs). The timing requirement of the ADV# pin assertion is different for these SRAMs, and this is enabled by setting SYSCFG 17h[1] = 1 (i.e., enabling pipelined synchronous SRAM). Table 4-21 lists the registers provided for SRAM support.

Table 4-18 Data SRAM Synchronous Configurations - Typical

Cache Size	Qty	Size
256K Bytes	2	32Kx32
512K Bytes	4	64Kx18

Table 4-19 SRAM Cycle Lengths

	Sync SRAMs	
Speed	**Standard**	**Pipelined Burst***
Read Burst Cycles		
50MHz	2-1-1-1	2-1-1-1 1-1-1-1
60MHz	3-1-1-1	3-1-1-1 1-1-1-1
66MHz	3-1-1-1	3-1-1-1 1-1-1-1
Write Burst Cycles		
50MHz	2-1-1-1	2-1-1-1 1-1-1-1
60MHz	3-1-1-1	3-1-1-1 1-1-1-1
66MHz	3-1-1-1	3-1-1-1 1-1-1-1

*This timing is for a single-bank. Leadoff cycle will be increased by one clock for a double-bank solution when it is a pipelined cycle due to the turn-around time of two banks and to prevent data contention between the banks.

Table 4-20 Tag and Data SRAM Speed Requirements

Para.	Description	50 MHz	60 MHz	66 MHz
Sync SRAM Data				
tCD	Clock Access Time	12ns (2-1-1-1)/ 12ns (3-1-1-1)	9ns	9ns
SRAM Tag for Sync Cache System				
tAA	Address Access Time	10ns (2-1-1-1)/ 20ns (3-1-1-1)	15ns	12ns

Table 4-21 Register Bits Associated with SRAM Support

7	6	5	4	3	2	1	0
SYSCFG 02h			**Cache Control Register 1**				**Default = 00h**
L2 cache size selection: **If SYSCFG 0Fh[0] = 0** 00 = 64KB 01 = 128KB 10 = 256KB 11 = 512KB **If SYSCFG 0Fh[0] = 1** 00 = 1MB 01 = Reserved 10 = Reserved 11 = Reserved		L2 cache write policy: 00 = L2 cache write-through 01 = Adaptive writeback Mode 1 10 = Adaptive writeback Mode 2 11 = L2 cache writeback		L2 cache operating mode select: 00 = Disable 01 = Test Mode 1; External Tag Write (Tag data write-through SYSCFG 07h) 10 = Test Mode 2; External Tag Read (Tag data read from SYSCFG 07h) 11 = Enable L2 cache		DRAM posted write: 0 = Disable 1 = Enable	CAS precharge time: 0 = 2 CPUCLKs 1 = 1 CPUCLK
SYSCFG 03h			**Cache Control Register 2**				**Default = 00h**
Timing for burst writes to L2 cache: 00 = X-4-4-4 01 = X-3-3-3 10 = X-2-2-2 11 = X-1-1-1		Leadoff cycle time for writes to L2 cache: 00 = 5-X-X-X 01 = 4-X-X-X 10 = 3-X-X-X 11 = 2-X-X-X		Timing for burst reads to L2 cache: 00 = X-4-4-4 01 = X-3-3-3 10 = X-2-2-2 11 = X-1-1-1		Leadoff cycle time for reads to L2 cache: 00 = 5-X-X-X 01 = 4-X-X-X 10 = 3-X-X-X 11 = 2-X-X-X	
SYSCFG 04h			**Shadow RAM Control Register 1**				**Default = 00h**
				Sync SRAM pipelined read cycle 1-1-1-1 enable:(1) 0 = Implies leadoff T-state for read pipe-lined cycle = 2(2) 1 = Enables leadoff T-state for read pipe-lined cycle = 1(3)			
(1) If SYSCFG 03h[3:2] = 11, then this register setting is valid. (2) It will be a 3-1-1-1 cycle followed by a 2-1-1-1 cycle, or a 3-1-1-1 cycle for successive pipelined cycles, based on SYSCFG 10h[5]. (3) It will be a 3-1-1-1 cycle followed by a 1-1-1-1 cycle for successive pipelined cycles. SYSCFG 10h[5] must be set to 1.							
SYSCFG 0Fh			**PCI Master Burst Control Register 2**				**Default = 00h**
							Cache size selection: This bit along with SYSCFG 02h[1:0] defines the L2 cache size. 0 = < 1MB 1 = 1MB

Table 4-21 Register Bits Associated with SRAM Support (cont.)

7	6	5	4	3	2	1	0
SYSCFG 10h			**Miscellaneous Control Register 1**				**Default = 00h**
		Leadoff cycle for a pipelined read: 0 = 3-X-X-X read followed by a 3-X-X-X pipelined read cycle 1 = 3-X-X-X read followed by a 2-X-X-X pipelined read cycle					
SYSCFG 11h			**Miscellaneous Control Register 2**				**Default = 00h**
					Page miss posted write: 0 = Enable 1 = Disable		
SYSCFG 17h			**PCI Cycle Control Register 2**				**Default = 00h**
						Sync SRAM type (if SYSCFG 11h[3] = 1): 0 = Standard 1 = Pipelined	

4.5.5.3 Sony SONIC-2WP (Cache Module) Support

The Sony SONIC-2WP is a single chip, writeback cache subsystem that integrates 256KB of cache memory, tag RAM and all other associated control logic. The integrated 256KB cache is direct-mapped and it supports 3-1-1-1 burst cycles, and operates at 3.3V. If this chip is used, SYSCFG 00h[5] should be set to 1. This causes a few changes in the signal functions of FireStar. The TAG1 and the TAG2 signals are connected to the START# signal from the Sony cache module. This signal is asserted by the Sony cache module when a CPU cycle translates to a read miss, write miss, or a write-through cycle. The assertion of this signal by the cache module causes FireStar to take control of the KEN# and BRDY# signals which it shares with the cache module. The TAG3 signal is connected to the BOFF# signal from the Sony cache module. The remainder of the TAG lines should be unconnected. All the other cache control signals of FireStar are not required and should be no connects. The ADS# input of FireStar should be connected to the SADS# output from the cache module. One note of caution, CPU pipelining must be disabled if using this cache module (i.e., set SYSCFG 02h[3:2] = 00).

Table 4-22 Cache Module Register Support

7	6	5	4	3	2	1	0
SYSCFG 00h		**Byte Merge/Prefetch & Sony Cache Module Control Register**					**Default = 00h**
		Sony SONIC-2WP support enable:(1) 0 = No Sony SONIC-2WP installed 1 = Sony SONIC-2WP installed					
(1) If bit 5 is set, ensure that the L2 cache has been disabled (i.e., set SYSCFG 02h[3:2] = 00).							

Table 4-23 SRAM Comparisons

Cycles	Sync	Pipelined Sync	Pipelined BSRAM	Sony Cache Module
Read hit	3-1-1-1	3-1-1-1	3-1-1-1	3-1-1-1
CPU pipelined RH	1-1-1-1	1-1-1-1	1-1-1-1	3-1-1-1(1)
2 BKs pipelined RH	1-1-1-1(2)	2-1-1-1(2)	2-1-1-1	3-1-1-1(1)
Write hit	3-1-1-1	3-1-1-1	3-1-1-1	3-1-1-1
Writeback	N	N+4	N+4	N+BOFF
PCI read	x-2-2-2	x-3-3-3	x-3-3-3	x-2-2-2(3)
PCI write	x-2-2-2	x-2-2-2	x-2-2-2	x-2-2-2(3)
Cost	High	Low	Low	High

1) No CPU pipelined for Sony Cache Module.
2) Data bus conflict for sync. SRAM, minimum data bus conflict for pipelined SRAM with 82C700 OE# control.
3) L2 needs “castout” dirty line before master access.

4.6 DRAM Controller

The DRAM controller within FireStar uses a 64-bit wide DRAM data bus interface. It also uses the page mode technique for faster data access from the DRAMs.

Page mode is always used in FireStar for CPU accesses, both for bursts and between bursts. Page mode is performed by keeping RAS active while reading or writing multiple words within a DRAM page by changing only the column address and toggling CAS with the new column address. The DRAM page size is fixed at 4KB.

Non-ISA refresh is used to increase the CPU bandwidth by not having to put the CPU on hold every 15μs to refresh the DRAM. The DRAM can be refreshed in the background while the CPU is accessing the internal cache.

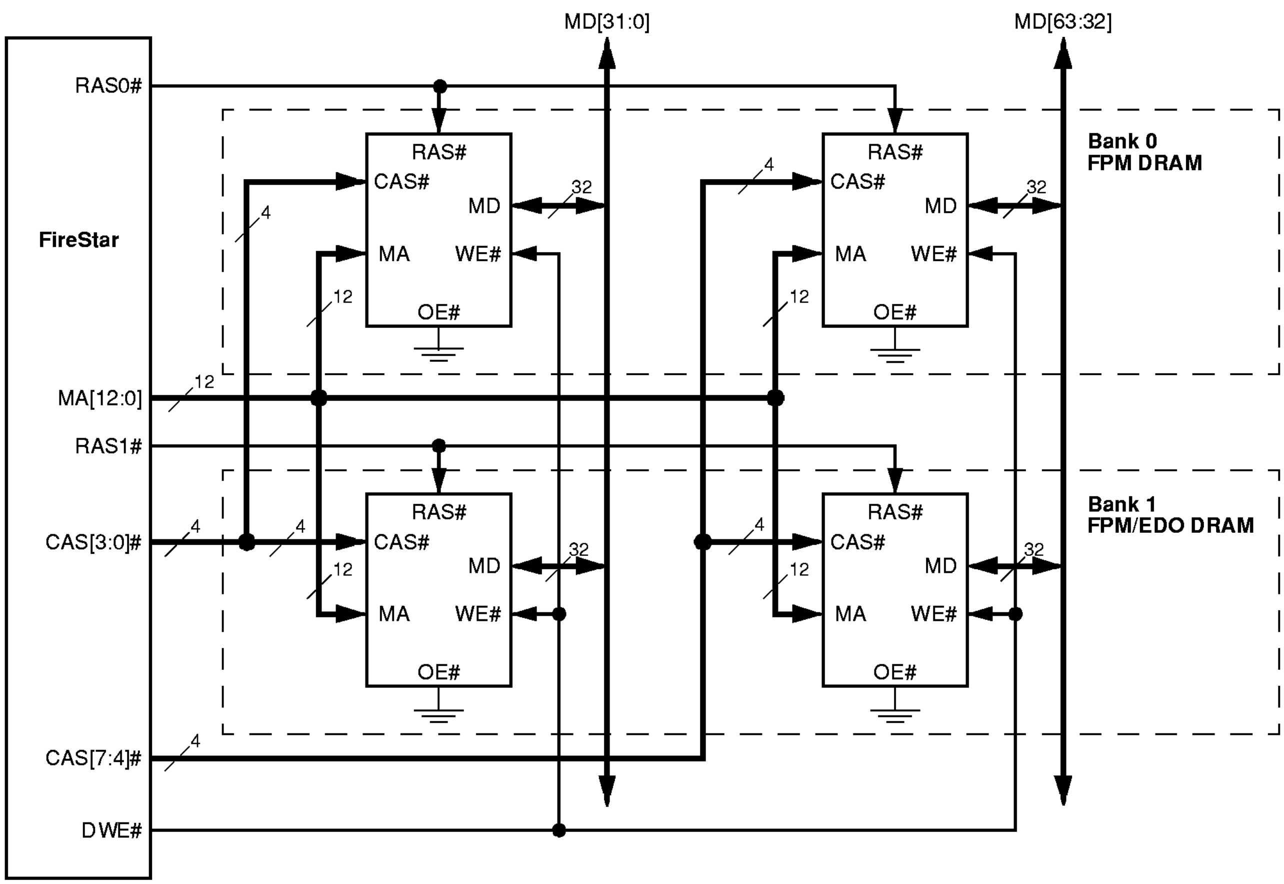
Asymmetric as well as **symmetric** DRAM sizes are supported.

- Banks 0-3 support:
  - 12x12, 12x11, 12x10, 12x9, 12x8, 11x11, 11x10, 11x9, 11x8, 10x10, 10x9, 10x8, and 9x9 DRAMs.
- Banks 4-5 support:
  - 12x12, 12x11, 11x11, 11x10, 10x10, 10x9, and 9x9 DRAMs only.
- MA12 Support:
  - In place of either of the RAS3# or RAS4# pin

4.6.1 FPM and EDO DRAM Support

The DRAM controller supports Fast Page Mode (FPM) and Extended Data Out (EDO) DRAMs. It can also be configured to support FPM and EDO DRAMs in different banks at the same time. Figure 4-13 gives the connectivity information for FPM DRAM.

Figure 4-13 FPM DRAM Connectivity



4.6.1.1 DRAM Type Detection Algorithm

FireStar can support both FPM and EDO DRAMs at the same time in different banks. The DRAM controller provides a mechanism by which the BIOS can easily determine the type of DRAM in each bank. After the CAS pulse is negated, EDO DRAMs continue to drive data out whereas FPM DRAMs will tristate their data buffers. If the bank is populated with FPM DRAM, and if the signal that is generated by the controller to latch the data from a location that is being read is delayed, the data that is read back will be incorrect. However, EDO DRAMs will still return the correct data. This principle is used to detect the type of DRAM in each bank. To test the DRAM type, the BIOS can set SYSCFG 1Fh[6] and write a data value to an address within the bank being tested, and read back the data from the same location. If the data that is read back is the same as the data written earlier, the bank contains EDO DRAM. If the data is different, the bank contains FPM DRAM.

After identifying the banks that are populated, the BIOS can use the following algorithm to determine the banks that are populated with EDO DRAM and program EDO timing for those banks. To perform type detection, EDO read timing has to be enabled (X-2-2-2).

To detect if a bank has EDO DRAM or FPM DRAM, follow these steps for each bank that is detected as non-empty in the DRAM sizing algorithm:

1. Set SYSCFG 14h[7] = 1 and PCIDV0 44h[0] = 1 (enables clocked mode in the DBC).
2. Set SYSCFG 1Ch[0] = 1 (global control for enabling X-2-2-2 burst timing).
3. Enable the bank to be tested by setting the size in SYSCFG 13h or 14h, and also by setting the asymmetricity bits in SYSCFG 24h. Disable all other banks.
4. Set the appropriate bit in SYSCFG 1Ch[7:2] (enable X-2-2-2 burst timing on a bank-by-bank basis; bit 7 for Bank 5, bit 2 for Bank 0).
5. Set SYSCFG 1Fh[6] = 1 (enable EDO DRAM testing).
6. Write an address at a 4KB boundary within the enabled bank (ADDR) with pattern 55555555h.
7. Write any other valid address with pattern 00000000h to clear the bus capacitance.
8. Read from address ADDR. If the data read back is 55555555h, the bank contains EDO DRAM. If the data that is read back is not 55555555h, the bank contains FPM DRAM. If the bank is detected as FPM DRAM, reset the corresponding bit in SYSCFG 1Ch[7:2].
9. Set SYSCFG 1Fh[6] = 0.
10. Continue from Step 3 and repeat all the steps until all the banks are identified.
11. Set SYSCFG 1Ch[0] = 0 to turn off X-2-2-2 burst read cycles. This bit can be turned on again for systems that use EDO DRAM.
12. Re-enable all the banks by programming the size registers SYSCFG 13h and/or SYSCFG 14h. Also program SYSCFG 24h for asymmetric DRAMs, and exit the routine. At this time, SYSCFG 1Ch[7:2] will be programmed to set the type of DRAM in each bank.

After identifying the DRAM type in each bank, the read and write timing must be programmed.

Table 4-24 DRAM Detection Register Bits

7	6	5	4	3	2	1	0
SYSCFG 14h			**Memory Decode Control Register 2**				**Default = 00h**
Data buffer control during configuration cycles: 0 = Normal 1 = Generate internal HDOE# signal Must = 1 for EDO timing.	Full decode for logical Bank 3 (RAS3#): 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)			SMRAM control: Inactive SMIACT#: 0 = Disable SMRAM 1 = Enable SMRAM(1) Active SMIACT#: 0 = Enable SMRAM for both Code and Data(1) 1 = Enable SMRAM for Code only(1)	Full decode for logical Bank 2 (RAS2#): 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)		

(1) If SYSCFG 13h[3] is set.

Table 4-24 DRAM Detection Register Bits (cont.)

7	6	5	4	3	2	1	0
PCIDV0 44h			**Data Path Register 1**				**Default = 00h**
							Memory read accesses in the DBC: 0 = FP Mode 1 = EDO/ SDRAM
SYSCFG 1Ch			**EDO DRAM Control Register**				**Default = 00h**
Bank 5: 0 = FPM DRAM 1 = EDO DRAM	Bank 4: 0 = FPM DRAM 1 = EDO DRAM	Bank 3: 0 = FPM DRAM 1 = EDO DRAM	Bank 2: 0 = FPM DRAM 1 = EDO DRAM	Bank 1: 0 = FPM DRAM 1 = EDO DRAM	Bank 0: 0 = FPM DRAM 1 = EDO DRAM		CAS pulse width during DRAM accesses: 0 = CAS pulse width determined by SYSCFG 01h[3] 1 = CAS pulse width is 1 CPUCLK(2)
(2) The width of the pulse is one CPUCLK for read accesses to banks that are populated with EDO DRAMs (selected by bits [7:2]), resulting in X-2-2-2 burst to EDO DRAM at 50/60/66MHz. SYSCFG 14h[7] and PCIDV0 44h[0] must be set in prior to setting this bit. X-2-2-2 burst cycles enabled by this bit apply only during CPU read bursts to EDO DRAM banks that are enabled in SYSCFG 1Ch[7:2].							
SYSCFG 13h			**Memory Decode Control Register 1**				**Default = 00h**
Reserved	Full decode for logical Bank 1 (RAS1#): 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)			SMRAM: 0 = Disable 1 = Enable See SYSCFG 14h[3]	Full decode for logical Bank 0 (RAS0#): 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)		
SYSCFG 1Fh			**EDO Timing Control Register**				**Default = 00h**
	0 = Normal (fast page mode) 1 = Detect EDO						

4.6.2 EDO Support

FireStar provides the capability to use EDO DRAMs in a system. EDO devices are very similar to devices that incorporate FPM accesses. However, the use of EDO DRAMs boosts the system performance considerably over conventional FPM DRAMs. This boost in performance stems from the different way in which the memory bus is controlled.

In conventional FPM DRAMs, the memory bus is turned on by the falling edge of CAS# and is turned off (High-Z) when CAS# returns to high. The FPM DRAMs only guarantee data to be valid for 5ns (which is typically too brief for systems operating at full speed). To compensate, CAS# must be held low for an extended period until the data can be read from the bus.

In contrast, EDO devices turn on the memory bus when CAS# falls low but do not turn off the bus when CAS# returns to high. Instead, the data remains valid until the next falling edge of CAS#. Because the data remains valid until the falling edge of the next CAS#, the transfer of memory data to the latch in the memory controller can be overlapped with the next column precharge. This extra time that the data remains valid resolves the system problem described above.

The extended data time allows the system to run with a minimum CAS# low time. This increases the system performance by decreasing the page access cycle time. Control of the memory bus can be obtained by the OE# and CAS# signals.

FireStar allows the user to populate the system with up to six banks of EDO DRAMs. Individual bits in SYSCFG 1Ch need to be set to a “1” for each bank that uses EDO DRAMs. Timing can be programmed to achieve a 6-2-2-2 read cycle at 50MHz when EDO DRAMs are used with FireStar.

FireStar also provides the flexibility to mix and match EDO DRAM SIMMs and conventional FPM DRAMs among the different banks. As an example, EDO DRAMs in Banks 0 and 2 could be used and FPM DRAMs in the other banks. There are no restrictions in terms of which bank(s) can contain EDO SIMMs and which can contain FPM DRAMs. However, care must be taken to ensure that the SIMMs that make-up the 64-bit data path to DRAM are all EDO DRAMs or all FPM DRAMs.

In a system, all banks that have been populated with EDO DRAMs will have the same DRAM timings. Likewise, all banks that are populated by FPM DRAMs will have the same DRAM timings.

Table 4-25 EDO Associated Register Bits

7	6	5	4	3	2	1	0
SYSCFG 14h			**Memory Decode Control Register 2**				**Default = 00h**
Data buffer control during configuration cycles: 0 = Normal 1 = Generate internal HDOE# signal Must = 1 for EDO timing.							
SYSCFG 1Ch			**EDO DRAM Control Register**				**Default = 00h**
Bank 5: 0 = FPM DRAM 1 = EDO DRAM	Bank 4: 0 = FPM DRAM 1 = EDO DRAM	Bank 3: 0 = FPM DRAM 1 = EDO DRAM	Bank 2: 0 = FPM DRAM 1 = EDO DRAM	Bank 1: 0 = FPM DRAM 1 = EDO DRAM	Bank 0: 0 = FPM DRAM 1 = EDO DRAM		CAS pulse width during DRAM accesses: 0 = CAS pulse width determined by SYSCFG 01h[3] 1 = CAS pulse width is 1 CPUCLK(2)
(2) The width of the pulse is one CPUCLK for read accesses to banks that are populated with EDO DRAMs (selected by bits [7:2]), resulting in X-2-2-2 burst to EDO DRAM at 50/60/66MHz. SYSCFG 14h[7] and PCIDV0 44h[0] must be set in prior to setting this bit. X-2-2-2 burst cycles enabled by this bit apply only during CPU read bursts to EDO DRAM banks that are enabled in SYSCFG 1Ch[7:2].							

Table 4-25 EDO Associated Register Bits (cont.)

7	6	5	4	3	2	1	0
SYSCFG 1Fh			**EDO Timing Control Register**				**Default = 00h**
0 = Normal 1 = Generate conflict during EDO detection (bit 6 set) if necessary	0 = Normal (fast page mode) 1 = Detect EDO						
SYSCFG 26h			**UMA Control Register 1**				**Default = 00h**
				5-2-2-2 EDO DRAM read timing at 66MHz in a cacheless system: 0 = Disable 1 = Enable			
SYSCFG 27h			**Miscellaneous Control Register 4**				**Default = 00h**
Master to EDO DRAM read cycle controlled by DWE#: 0 = Disable 1 = Enable							
PCIDV0 44h			**Data Path Register 1**				**Default = 00h**
				Memory read accesses in the DBC If PCIDV0 44h[0] = 1 and 47h[7] = 1: 0 = SDRAM 1 = Reserved			Memory read accesses in the DBC: 0 = FP Mode 1 = EDO/ SDRAM
PCIDV0 45h			**Data Path Control Register 2**				**Default = 00h**
		Ping-pong buffer reset of CPU read EDO is qualified with HDOE#. 0 = Disable 1 = Enable					
PCIDV0 47h			**Data Path Control Register 4**				**Default = 00h**
SDRAM memory read accesses in DBC: 0 = Disable 1 = Enable							

4.6.3 SDRAM Support

FireStar provides the capability to use SDRAM DIMM modules in a system design. Up to four banks of SDRAM banks are supported. The SDRAM devices accept all its input command signals at the rising edge of system clock. The clocking allows data pipelining within the SDRAM device and data output in a continuous stream on every clock. Because of this, an SDRAM-based design boosts the memory performance considerably over FPM/EDO DRAMs. Table 4-26 gives a summary of SDRAM cycle lengths.

4.6.3.1 SDRAM Commands

With SDRAM, external control signals are latched with the rising edge of clock pulses and specific high and low combinations are recognized as commands. The SDCS# (SDCS[3:0]#), SDRAS#, SDCAS#, SDWE#, and MA address lines define the inputs commands which become active on the positive transition of SDCKE.

FireStar issues the following SDRAM COMMANDs:
- BANK ACTIVE
- BANK PRECHARGE
- PRECHAGE ALL
- WRITE
- READ
- MODE REGISTER SET
- AUTO REFRESH

FireStar does not support the following commands:
- READ WITH AUTOPRECHARGE
- WRITE WITH AUTOPRECHARGE
- CLOCK SUSPEND MODE
- SELF REFRESH

SDRAM READ and WRITE Commands

To avoid bus contention on the memory bus, FireStar ensures a dead cycle between write data and read data commands. During read cycles, SDDQM[7:0]# are used as the standard output enable function. During write cycles, these outputs act as a mask for input data buffer of the SDRAM.

4.6.3.2 SDRAM Initialization

FireStar allows SDRAM devices to stabilize and will not toggle any input command signal for 100ms. The first command will be the PRECHARGE ALL banks. After precharging, the mode register will be programmed based on the register setting. The following fields will be affected by this programming:
- Burst length: 1, 2, 4, 8 or full page
- Wrap Type: Sequential or interleaved
- CAS# latency: 1, 2 or 3

SDRAM refreshing is done in the similar fashion of FPM and EDO DRAMs. Burst refresh should never be enabled when SDRAM is being used in the system.

Table 4-27 shows the register bits associated for configuring SDRAM in a FireStar-based system.

4.6.3.3 Unbuffered DIMMs

FireStar supports up to four banks of unbuffered SDRAM DIMMs connectors on a system motherboard. The maximum clock frequency is 66MHz. Clock should be connected to each of the DIMM connector.

MA[10:0] are connected as the row address of the DIMM. MA11 is used as the bank select pin. SDCS# pins are used as the bank select outputs.

Table 4-26 SDRAM Cycle Lengths

CPU Bus Speed	Page Hit Leadoff	Page Miss No Precharge	Page Miss Precharge
Read Burst Cycle			
66MHz, CAS Latency = 3	7 cycles	10 cycles	13 cycles
66MHz, CAS Latency = 2	6 cycles	9 cycles	12 cycles
Write Burst Cycle			
From ADS# to WRITE command	5 cycles	8 cycles	11 cycles

Table 4-27 SDRAM Configuration Registers

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">SYSCFG 0Eh</td><td colspan="4">PCI Master Burst Control Register 1</td><td colspan="2">Default = 00h</td></tr>
<tr><td></td><td>ISA/DMA master through internal MRD#/ MWR#:
0 = Disable
1 = Enable
This bit must be turned off for DMA support with SDRAM.</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td colspan="2">SYSCFG 28h</td><td colspan="4">SDRAM Control Register 1</td><td colspan="2">Default = 00h</td></tr>
<tr><td></td><td colspan="3">SDRAM CAS# latency:
001 = 1
010 = 2
011 = 3
All other combinations = Reserved</td><td>Write-through:
0 = Sequential
1 = Interleaved</td><td colspan="3">SDRAM burst length control:
000 = 1
010 = 4
All other combinations = Reserved</td></tr>
<tr><td colspan="2">SYSCFG 29h</td><td colspan="4">SDRAM Control Register 2</td><td colspan="2">Default = 00h</td></tr>
<tr><td></td><td></td><td colspan="2"></td><td>Bank 3 SDRAM:
0 = Disable
1 = Enable</td><td>Bank 2 SDRAM:
0 = Disable
1 = Enable</td><td>Bank 1 SDRAM:
0 = Disable
1 = Enable</td><td>Bank 0 SDRAM:
0 = Disable
1 = Enable</td></tr>
<tr><td colspan="2">SYSCFG 2Eh</td><td colspan="4">UMA Control Register 2</td><td colspan="2">Default = 00h</td></tr>
<tr><td>Allow SDRAM self-refresh in Suspend mode:
0 = Disable (SDRAM engages auto-refresh mode)
1 = Enable (need to enable SDRAM self-refresh if SYSCFG 12h[5:4] = 01 or 10)</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

Table 4-27 SDRAM Configuration Registers (cont.)

7	6	5	4	3	2	1	0
PCIDV0 44h			**Data Path Register 1**				**Default = 00h**
				Memory read accesses in the DBC If PCIDV0 44h[0] = 1 and 47h[7] = 1: 0 = SDRAM 1 = Reserved			Memory read accesses in the DBC: 0 = FP Mode 1 = EDO/ SDRAM
PCIDV0 47h			**Data Path Control Register 4**				**Default = 00h**
SDRAM memory read accesses in DBC: 0 = Disable 1 = Enable							
PCIDV0 48h			**Data Path Control Register 5**				**Default = 00h**
		During refresh cycles if this bit = 1, the RAS# corresponding to the bank with size 0 will not be generated for SDRAM.			SDRAM mode select 000 = Normal SDRAM mode 001 = NOP command enable 010 = All banks precharge 011 = Mode register command enable 100 = CBR cycle enable All other combinations = Reserved		
PCIDV0 4Eh			**SDRAM Control Register**				**Default = 00h**
SDRAM + L2 + pipelining:(1) 0 = Disable 1 = Enable	SDRAM + L2 + pipelining:(1) 0 = Disable 1 = Enable	6 CLK SDRAM leadoff: 0 = Disable 1 = Enable	Reserved				
(1) Bits 7 and 6 have been implemented to solve two problems with the DIRTY signal in systems that have SDRAM + L2 + pipelining.							

4.6.3.4 SDRAM Detection Algorithm

The following steps detail (in chronological order) the FPM/EDO/SDRAM detection algorithm for FireStar.

1. Power on.
2. Wait for 200 µs. Ensure that L1 and L2 cache are off.
3. If any of the RAS lines are to be mapped as PIO or alternate function, program the associated registers at this time.
4. Set SYSCFG 14h[7] = 1, and PCIDV0 44h[0] = 1 in order to enable the clocked mode of operation in the DBC (Data Buffer Controller) module.
5. Set SYSCFG 28h according to the following options to set up the operating mode parameters for SDRAM operation. Also set SYSCFG 29h = 00h.

   For CAS Latency = 3, interleaved burst (Intel), set SYSCFG 28h = 3Ah (Default)

   For CAS Latency = 3, linear burst (Cyrix), set SYSCFG 28h = 32h

   For CAS Latency = 2, interleaved burst (Intel), set SYSCFG 28h = 2Ah

   For CAS Latency = 2, linear burst (Cyrix), set SYSCFG 28h = 22h

6a. Enable the CPU-to-DRAM buffer without byte merge:
SYSCFG 01h[2] = 1
SYSCFG 02h[1:0] = 11
SYSCFG 2Ch[0] = 1
PCIDV0 44h[4] = 1

6b. Also set the following bits to enable read-around:
SYSCFG 2Ch[5] = 1
PCIDV0 45h[5:4] = 11

Note: Step 6a must be performed while testing SDRAM. SDRAM will not work if the CPU-to-DRAM buffer is turned off.

If SDRAM is present in the system, step 6b must be performed before testing the L2 cache. SDRAM will not work with L2 cache if step 6B is not performed.

7. Set PCIDV0 48h[2:0] = 010 for bank precharge command for SDRAM banks.
8. Set X = 0 (X is bank count variable).
9. Set the size corresponding to Bank X in SYSCFG 13h[6:4], 13h[2:0], 14h[6:4], and 14h[2:0] to 2MB. Set the size for all other banks to 0MB. Set the DRAM type for Bank X as “SDRAM” in SYSCFG 29h[3:0], and also set the DRAM type for the bank as “EDO” in SYSCFG 1Ch[5:2].
10. Read memory location 0h in order to precharge all open pages in Bank X, if Bank X is populated with SDRAM.
11. Set X = X + 1.
12. If X = 3, go back to Step 9. If X = 4, set the size for all banks as 0MB, SYSCFG 29h[3:0] = 0000, and SYSCFG 1Ch[7:2] = 000000. Continue with Step 13.
13. Set SYSCFG 29h[3:0] = 0000, and SYSCFG 1Ch[5:2] = 0000 for Fast Page Mode DRAM operation only.
14. Set PCIDV0 48h[5] = 1 to disable RASx# generation to a bank with 0 MB during refresh cycles.
15. Enable chosen refresh mode.
16. Wait for eight refresh periods and then set PCIDV0 48h[2:0] = 000.
17. Follow the algorithm on page 86 for Banks 0-5. Store the information, without programming the FireStar registers. Reset all the bank sizes to 0MB after completing the detection. Also set SYSCFG 29h[3:0] = 0000, and SYSCFG 1Ch[7:2] = 000000. If all the banks are identified with either Fast Page Mode or EDO DRAM, skip to Step 29. Else, continue with Step 18.
18. Disable refresh (in sequence):
PCIDV1 47h[6] = 0
PCIDV1 64h[0] = 1
SYSCFG 2Eh[6] = 0
SYSCFG 2Fh[2:0] = 000
SYSCFG 27h[2:0] = 000
19. Set PCIDV0 47h[7] = 1 to enable the SDRAM data path in the DBC module.
20. Set the DRAM type only for bank not detected with either FPM DRAM or EDO DRAM, to “SDRAM” in SYSCFG 29h[3:0] and SYSCFG 1Ch[5:2]. Also set the size corresponding to the current bank to 2MB in SYSCFG 13h[6:4], 13h[2:0], 14h[6:4], or 14h[2:0]. Set the bank size for all other banks to 0MB.
21. Set PCIDV0 48h[2:0] = 001 to enable NOP command. Read from address 0h to force the current SDRAM bank to the NOP state.
22. Set PCIDV0 48h[2:0] = 100 for SDRAM refresh mode. Read from address 0h eight times.
23. Set PCIDV0 48h[2:0] = 011 to enable Mode Register Set command.
24. Read from the following addresses to load the 3CLK/2CLK CAS latency, interleaved/linear access, and burst length of 4 information into the current SDRAM bank.

    For CAS Latency = 3, interleaved burst (Intel), read from address 000001D0h (Default)

For CAS Latency = 3, linear burst (Cyrix), read from address 00000190h

For CAS Latency = 2, interleaved burst (Intel), read from address 00000150h

For CAS Latency = 2, linear burst (Cyrix), read from address 00000110h

25. Set PCIDV0 48h[2:0] = 100 for SDRAM refresh mode. Read from address 0h eight times.
26. Set PCIDV0 48h[2:0] = 000 to enable normal SDRAM mode.
27. Detect SDRAM presence or absence on the current DRAM bank by writing a known pattern to an address within the size enabled for this bank, reading back from the same address, and comparing the size.
28. Store the SDRAM presence or absence information. If any non-FPM/EDO banks remain, go to Step 20. Else, program all the bank sizes to 0MB, and continue with Step 29.

29a. Retrieve information about the presence or absence of FPM/EDO/SDRAM in each of the banks and program the following registers accordingly:

For banks detected with FPM DRAM, set the corresponding bits in SYSCFG 29h[3:0] and SYSCFG 1Ch[7:2] = 0.

For banks detected with EDO DRAM, set the corresponding bits in SYSCFG 29h[3:0] = 0, and SYSCFG 1Ch[7:2] = 1.

For banks detected with SDRAM, set the corresponding bits in SYSCFG 29h[3:0] = 1, and SYSCFG 1Ch[7:2] = 1.

29b. If all the banks are FPM/EDO DRAM, the CPU-to-DRAM buffer may be turned off, and may again be enabled with or without DRAM byte merge based on the setup option. The read-around feature can also be turned off and may be enabled based on the setup option. If none of the banks are SDRAM, set PCIDV0 47h[7] = 0.

To turn off the CPU-to-DRAM buffer, set the following registers:
PCIDV0 44h[4] = 0
SYSCFG 2Ch[0] = 0

To turn off the read-around feature, follow these steps:
PCIDV0 45h[5:4] = 00
SYSCFG 2Ch[5] = 0

If SDRAM is detected in any of the banks, the CPU-to-DRAM buffer, and the read-around feature MUST NOT be turned off. Also, on the current silicon revision, if SDRAM is detected, DRAM byte merge MUST NOT be turned on, even if enabled in the setup.

30. Set PCIDV0 48h[2:0] = 000 for normal SDRAM mode.
31. Enable chosen refresh mode.
32. Wait for eight refresh periods.
33. For the banks populated with FPM or EDO DRAM, follow the algorithm page 86 to detect the size. For banks populated with SDRAM, detect the size by using standard procedure.
34. Program the size information in the FireStar registers and exit.

Note: SDRAM is only supported on Banks 0-3. Bank 4 can only be FPM/EDO DRAM. Bank 4 support can be enabled by setting SYSCFG 19h[3] = 1, and by programming SYSCFG 19h[2:0] for size. Bank 5 can only be enabled by setting SYSCFG 19h[7] in which case L2 cache cannot be supported. Bank 5 size information can be programmed in SYSCFG 19h[6:4].

4.6.4 DRAM Buffering

Deep buffering is one of the major performance enhancements in FireStar. It incorporates deep buffers in the CPU-to-DRAM and PCI-to-DRAM data paths, which enables read prefetching and write posting in the data paths.

- Deep buffering for DRAM performance
  - Six quad-word CPU-to-DRAM write posting
  - 24 double-word PCI-to-DRAM write posting
  - 24 double-word DRAM-to-PCI read prefetch

Table 4-28 lists the registers/bits that are associated with DRAM Buffering.

4.6.4.1 CPU-to-DRAM Deep Buffer

A six quad-word deep FIFO is built into the DBC in the CPU-to-DRAM path. These deep buffers are used by FireStar to buffer CPU-to-DRAM data. Once the DRAM is free, the buffered data is dispatched into the DRAM. This way the system level latencies caused by shared resources are minimized.

4.6.4.2 PCI-to-DRAM Deep Buffer

A 24 double-word buffer has been designed into the PCI-to-DRAM path. During PCI master write bursts, the master posts data into this buffer. Once CPU-to-DRAM cycles are completed, the posted data will be written back in to the DRAM. This avoids any stalling in the PCI and full bus bandwidth is utilized.

Table 4-28 DRAM Buffering Related Registers/Bits

7	6	5	4	3	2	1	0
PCIDV0 44h			**Data Path Register 1**				**Default = 00h**
FIFO for CPU write to PCI: 0 = Disable 1 = Enable	FIFO for PCI read from DRAM: 0 = Disable 1 = Enable	FIFO for PCI write to DRAM: 0 = Disable 1 = Enable	FIFO for CPU write to DRAM: 0 = Disable 1 = Enable				
SYSCFG 2Ah			**PCI-to-DRAM Control Register 1**				**Default = 00h**
			PCI TRDY# wait state control with PCI-to-DRAM deep buffer: 0 = 0WS (X-1-1-1) 1 = 1WS (X-2-2-2)	Write burst with PCI-to-DRAM deep buffer: 0 = Disable 1 = Enable	Read burst with PCI-to-DRAM deep buffer: 0 = Disable 1 = Enable		PCI-to-DRAM deep buffer size: 0 = 16 dword 1 = 24 dword
SYSCFG 2Ch			**CPU-to-DRAM Buffer Control Register**				**Default = 00h**
CPU-to-PCI read and CPU-to-DRAM write concurrency: 0 = Disable 1 = Enable	This bit needs SYSCFG 2Ch[5] to be enabled. When set (to 1), the cache write to CPU-to-DRAM buffer becomes more aggressive. Will save approximately 3 clocks over the previous method. Not supported.	When set (to 1) along with CPU-to-DRAM buffer and PBSRAM, the DRAM controller will first supply the data to the CPU before writing the previous data back to DRAM during a cache miss dirty cycle.(1)	CPU-to-PCI write and CPU-to-DRAM read concurrency: 0 = Disable 1 = Enable	Enable internal LMEM# during special cycles: 0 = No 1 = Yes	BOFF# assertion during DRAM read cycles: 0 = Disable 1 = Enable	Data merging when CPU owns DRAM bus: 0 = Possible only when GUI owns DRAM bus (UMA feature - not supported) 1 = Always possible	Allow data collection while CPU-to-DRAM FIFO is flushing: 0 = Disable(2) 1 = Enable

(1) Bit 5's function needs the CPU-to-DRAM buffer to be enabled. DRAM processing for the read will start concurrently while data from cache will be written to the CPU-to-DRAM buffer.

(2) BOFF# is generated for the next DRAM write cycle as long as there is data in the FIFO.

4.6.5 Programming the DRAM Parameters

There are various parameters that can be obtained in the DRAM state machine - drive strengths, number of banks, bank size, DRAM type, and timing parameters.

4.6.5.1 Drive Strengths

Programmable current drive for the MA[12:0], RAS[5:0]# and the DWE# lines is provided. If SYSCFG 18h[6,4] = 10, then the current drive on these lines is 4mA (refer to Table 4-29). In this case, two F244 buffers will be required to drive each pair of DRAM banks. If SYSCFG 18h[6,4] = 01, then the current drive on these lines is increased to 16mA and it should be possible to drive the first pair of banks that are populated with DRAMs that are 4, 8, or 16 bits wide without any buffers.

Note that on the FireStar ACPI version the DWE line can be programmed for either 16mA or 20mA. These bits must be programmed prior to accessing any location in DRAM; they must be programmed before sizing the DRAM.

4.6.5.2 Number of DRAM banks

FireStar supports up to six banks of DRAM. The default condition is four banks of DRAM supporting up to 512Mbytes of system memory.

RAS4# and RAS5# are selected through SYSCFG 19h[7] (RAS5#), and 19h[3] (RAS4#). Table 4-30 shows the bits control full memory decode and the RAS selection bits.

Table 4-29 Drive Strength Control Bits

7	6	5	4	3	2	1	0
SYSCFG 18h			**Interface Control Register**				**Default = 00h**
Reserved	Drive strength on RAS lines: 0 = 16mA 1 = 4mA	CAS lines voltage selection: 0 = 5.0V 1 = 3.3V	Drive strength on memory address lines and write enable line: 0 = 4mA 1 = 16mA				
SYSCFG 18h - FS ACPI Version			**Interface Control Register**				**Default = 00h**
Drive strength on SDRAS and SDCAS lines: 0 = 16mA 1 = 4mA	Drive strength on RAS lines: 0 = 16mA 1 = 4mA	CAS lines voltage selection: 0 = 5.0V 1 = 3.3V	Drive strength on memory address lines: 0 = 4mA 1 = 16mA	Drive strength on write enable line: 0 = 16mA 1 = 20mA			

Table 4-30 RAS Selection Bits

7	6	5	4	3	2	1	0
SYSCFG 19h			**Memory Decode Control Register 3**				**Default = 00h**
Pin functionality: 0 = GWE# 1 = RAS5#	Full decode for logical Bank 5 (RAS5#) if SYSCFG 19h[7] is set: 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)			Bank 4 (RAS4#): 0 = Disable 1 = Enable	Full decode for logical Bank 4 (RAS4#): 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)		

4.6.5.3 DRAM Size

The DRAM bank size is set by programming groups of bits in SYSCFG 13h, 14h, 19h, and 24h. There is no required ordering for these selections: Any desired bank can be populated or skipped. For example, if in the course of testing system DRAM the BIOS POST code should find Bank 3 (RAS3#) defective, it should simply set that bank to "disabled." The DRAM controller will automatically map around it and provide a contiguous memory map to the system.

The bank size is determined by the number of rows and columns implemented by the DRAM chips that populate the bank. In this discussion, the term "symmetric" is used for DRAMs that have either an equal number of rows and columns, or one more row address line than the number of column address lines. The term "asymmetric" is used for DRAMs in which the number of row address lines exceed the number of column address lines by at least two. For example, DRAMs with 12 rows and 12 columns (12x12) are considered symmetric, DRAMs with 12 rows and 11 columns (12x11) are also considered symmetric. However, 12x10 DRAMs are considered asymmetric.

Banks 0-3 can be programmed to support symmetric or asymmetric DRAMs (12x12, 12x11, 12x10, 12x9, 12x8, 11x11, 11x10, 11x9, 11x8, 10x10, 10x9, 10x8, and 9x9). However, Banks 4-5 can only support symmetric DRAMs (12x12, 12x11, 11x11, 11x10, 10x10, 10x, and 9x9).

Table 4-31 shows the register bits that pertain to bank size. The DRAM size for Banks 0-3 is controlled by SYSCFG 13h-14h and SYSCFG 24h. However, the size for Banks 4-5 is controlled only by SYSCFG 19h. For Banks 0-3, if the correspond asymmetric bits in SYSCFG 24h are set to 00, and if the size is programmed in SYSCFG 13h or 14h, the DRAM controller will assume that the respective banks are populated with symmetric DRAMs.

Table 4-32 shows the register bits that need to be programmed for Banks 0-3 for various DRAM sizes.

Table 4-33 and Table 4-34 show the CPU address to memory address map. A3 and A4 must go through an internal burst counter, for the generation of the MA address to the DRAMs.

Table 4-31 DRAM Configuration Related Register Bits

7	6	5	4	3	2	1	0
SYSCFG 13h			**Memory Decode Control Register 1**				**Default = 00h**
	Full decode for logical Bank 1 (RAS1#): 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)				Full decode for logical Bank 0 (RAS0#): 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)		
SYSCFG 14h			**Memory Decode Control Register 2**				**Default = 00h**
	Full decode for logical Bank 3 (RAS3#): 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)				Full decode for logical Bank 2 (RAS2#): 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)		
SYSCFG 19h			**Memory Decode Control Register 3**				**Default = 00h**
	Full decode for logical Bank 5 (RAS5#) if SYSCFG 19h[7] is set: 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)				Full decode for logical Bank 4 (RAS4#): 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)		
SYSCFG 24h			**Asymmetric DRAM Configuration Register**				**Default = 00h**
Logical Bank 3 DRAM type: 00 = Sym DRAM 01 = Asym DRAM - x8 type 10 = Asym DRAM - x9 type 11= Asym DRAM - x10 type		Logical Bank 2 DRAM type: 00 = Sym DRAM 01 = Asym DRAM - x8 type 10 = Asym DRAM - x9 type 11= Asym DRAM - x10 type		Logical Bank 1 DRAM type: 00 = Sym DRAM 01 = Asym DRAM - x8 type 10 = Asym DRAM - x9 type 11= Asym DRAM - x10 type		Logical Bank 0 DRAM type: 00 = Sym DRAM 01 = Asym DRAM - x8 type 10 = Asym DRAM - x9 type 11= Asym DRAM - x10 type	

Table 4-32 Programming Size Registers

DRAM Size	Bank Size Setting in SYSCFG 13h or 14h	Bank Asymmetricity Setting in SYSCFG 24h
12x12	128MB	Symmetric
12x11	64MB	Symmetric
12x10	32MB	x10
12x9	16MB	x9
12x8	8MB	x8
11x11	32MB	Symmetric
11x10	16MB	Symmetric
11x9	8MB	x9
11x8	4MB	x8
10x10	8MB	Symmetric
10x9	4MB	Symmetric
10x8	2MB	x8
9x9	2MB	Symmetric

Table 4-33 CPU Address to Memory Row/Column Address Map

Mem Addr.	12x8 (8MB)		12x9 (16MB)		12x10 (32MB)		12x11 (64MB)		12x12 (128MB)		11x8 (4MB)		11x9 (8MB)		11x10 (16MB)		11x11 (32MB)	
	Col	Row	Col	Row	Col	Row	Col	Row	Col	Row	Col	Row	Col	Row	Col	Row	Col	Row
MA0	A3	A12	A3	A12	A3	A12	A3	A12	A3	A12	A3	A12	A3	A12	A3	A12	A3	A12
MA1	A4	A13	A4	A13	A4	A13	A4	A13	A4	A13	A4	A13	A4	A13	A4	A13	A4	A13
MA2	A5	A14	A5	A14	A5	A14	A5	A14	A5	A14	A5	A14	A5	A14	A5	A14	A5	A14
MA3	A6	A15	A6	A15	A6	A15	A6	A15	A6	A15	A6	A15	A6	A15	A6	A15	A6	A15
MA4	A7	A16	A7	A16	A7	A16	A7	A16	A7	A16	A7	A16	A7	A16	A7	A16	A7	A16
MA5	A8	A17	A8	A17	A8	A17	A8	A17	A8	A17	A8	A17	A8	A17	A8	A17	A8	A17
MA6	A9	A18	A9	A18	A9	A18	A9	A18	A9	A18	A9	A18	A9	A18	A9	A18	A9	A18
MA7	A10	A19	A10	A19	A10	A19	A10	A19	A10	A19	A10	A19	A10	A19	A10	A19	A10	A19
MA8	A11	A20	A11	A20	A11	A20	A11	A20	A11	A20	A11	A20	A11	A20	A11	A20	A11	A20
MA9	A22	A11	A22	A21	A22	A21	A22	A21	A22	A21	A22	A11	A22	A21	A22	A21	A22	A21
MA10	A24	A21	A24	A22	A24	A23	A24	A23	A24	A23	A24	A21	A24	A22	A24	A23	A24	A23
MA11	A26	A22	A26	A23	A26	A24	A26	A25	A26	A25	A26	A22	A26	A23	A26	A24	A26	A25

Note: Signals that are shaded are driven, but should be ignored by DRAM.

Table 4-34 CPU Address to Memory Row/Column Address Map

Addr.	10x8 (2MB) Col	10x8 (2MB) Row	10x9 (4MB) Col	10x9 (4MB) Row	10x10 (8MB) Col	10x10 (8MB) Row		9x9 (2MB) Col	9x9 (2MB) Row
MA0	A3	A12	A3	A12	A3	A12		A3	A12
MA1	A4	A13	A4	A13	A4	A13		A4	A13
MA2	A5	A14	A5	A14	A5	A14		A5	A14
MA3	A6	A15	A6	A15	A6	A15		A6	A15
MA4	A7	A16	A7	A16	A7	A16		A7	A16
MA5	A8	A17	A8	A17	A8	A17		A8	A17
MA6	A9	A18	A9	A18	A9	A18		A9	A18
MA7	A10	A19	A10	A19	A10	A19		A10	A19
MA8	A11	A20	A11	A20	A11	A20		A11	A20
MA9	A22	A11	A22	A21	A22	A21		A22	A21
MA10	A24	A21	A24	A22	A24	A23		A24	A22
MA11	A26	A22	A26	A23	A26	A24		A26	A23

Note: Signals that are shaded are driven, but should be ignored by DRAM.

DRAM Sizing Algorithm

This subsection describes the DRAM detection and sizing algorithm on FireStar. The algorithm will detect all the possible DRAM configurations listed in Table 4-32.

This discussion assumes that only Banks 0 through 3 are used in the system.

DRAM Detection and Sizing Algorithm

1. Turn L1 and L2 cache off.
2. Set i = 0 (bank number to be tested).
3. DRAM detection:
   - If i = 4, exit. If not, set the size for Bank i to be 128MB.
   - Set the size for all other banks as 0MB in SYSCFG 13h and 14h.
   - Write address 00000000h with pattern 55555555h.
   - Read from address 0000000h.
     - If the pattern that is read back is not 55555555h, Bank i does not contain any DRAM. Increment i, and go back to Step 3.
     - If the pattern that is read back is 55555555h, Bank i contains DRAM. Continue with Step 4.
4. Test of 128MB (12x12):
   - Bank i was set for 128MB in the previous step.
   - Write address 00000000h with pattern 55555555h.
   - Write address 04000000h (A26 = 1) with pattern AAAAAAAAh.
   - Read from address 00000000h.
     - If it is 55555555h, Bank i contains 128MB. Store this information, for updating the size registers 13h and 14h after exiting from this program. Increment i, and continue from Step 3.
     - If the pattern that is read back is not 55555555h, Bank i has less than 128MB. Continue with Step 5.
5. Test of 64MB (12x11):
   - Set the size for Bank i as 64MB. Set the size for all other banks as 0MB.
   - Write address 00000000h with pattern 55555555h.
   - Write address 01000000h (A24 = 1) with pattern AAAAAAAAh.
   - Write address 02000000h (A25 = 1) with pattern FEDCBA98h.
   - Read from address 00000000h.
     - If it is 55555555h, Bank i contains 64MB. Store this information, for updating the size registers 13h and 14h after exiting from this program. Increment i, and continue from Step 3.
     - If the pattern that is read back is not 55555555h, Bank i has less than 64MB. Continue with Step 6.

6A. Test of 32MB (11x11):
   - Set the size for Bank i as 32MB.
   - Set the size for all other banks as 0MB.
   - Write address 00000000h with pattern 55555555h.
   - Write address 01000000h (A24 = 1) with pattern AAAAAAAAh.
   - Read from address 00000000h.
     - If it is 55555555h, Bank i contains 11x11 32MB DRAM. Store this information, for updating the size registers 13h and 14h after exiting from this program. Increment i, and continue from Step 3.
     - If the pattern that is read back is not 55555555h, Bank i has either 12x10 32MB, or less than 32MB of DRAM. Continue with Step 6B.

6B. Test of 32MB (12x10):
   - The DRAM size was set as 32MB in Step 6A. Set the asymmetric bits corresponding to Bank i in SYSCFG 24h as x10.
   - Write address 00000000h with pattern 55555555h.
   - Write address 00400000h (A22 = 1) with pattern AAAAAAAAh.
   - Write address 01000000h (A24 = 1) with pattern FEDCBA98h.
   - Read from address 00000000h.
     - If it is 55555555h, Bank i contains 12x10 32MB DRAM. Store this information, for updating the size registers 13h and 14h after exiting from this program. Increment i, and continue from Step 3.
     - If the pattern that is read back is not 55555555h, Bank i has less than 32MB of DRAM. Continue with Step 7A.

7A. Test of 16MB (12x9):
   - Set the size for Bank i as 16MB.
   - Also set the asymmetric DRAM bits for Bank i in SYSCFG 24h as x9.
   - Write address 00000000h with 55555555h.
   - Write address 00000800h (A11 = 1) with pattern AAAAAAAAh.
   - Write address 00800000h (A23 = 1) with data FEDCBA98h.
   - Read from address 00000000h.
     - If it is 55555555h, Bank i has 12x9 DRAM. Store this information, for updating the size registers 13h and 14h after exiting from this program. Increment i, and continue from Step 3.
     - If the data that is read back is not 55555555h, continue with Step 7B.

7B. Test of 16MB (11x10):
   - Set the size for Bank i as 16MB.
   - Also set the asymmetric DRAM bits for Bank i in SYSCFG 24h as “00”.
   - Write address 00000000h with 55555555h.
   - Write address 00400000h (A22 = 1) with pattern AAAAAAAAh.

- Write address 00800000h (A23 = 1) with data FEDCBA98h.
- Read from address 00000000h.
  - If it is 55555555h, Bank i has 11x10 DRAM. Store this information, for updating the size registers 13h and 14h after exiting from this program. Increment i, and continue from Step 3.
  - If the data that is read back is not 55555555h, continue with Step 8A.

8A. Test of 8MB (12x8, 11x9, or 10x10):
- Set the size for Bank i as 8MB.
- Also set the asymmetric DRAM bits for Bank i in SYSCFG 24h as x8.
- Write address 00000000h with 55555555h.
- Write address 00400000h (A22 = 1) with data AAAAAAAAh.
- Read from address 00000000h.
  - If it is 55555555h, Bank i has 12x8 or 10x10 DRAM; continue from Step 8B.
  - If the data that is read back is not 55555555h, go to Step 8C.

8B. Distinguish between 12x8 and 10x10:
- The size for Bank i was set as 8MB, 12x8. Address 00000000h must still have 55555555h.
- Write address 00200000 with data AAAAAAAAh (A22 = 0, A21 = 1).
- Read from address 00000000h.
  - If it is pattern 55555555h, Bank i has 12x8 DRAM. Store this information, for updating the size registers 13h and 14h after exiting from this program. Increment i and continue with Step 3.
  - If the data that is read back is not 55555555h, Bank i has 10x10 DRAM. Store this information, for updating the size registers 13h and 14h after exiting from this program. Reset the bits corresponding to Bank i in SYSCFG 24h to 00b. Increment i and continue with Step 3.

8C. Test of 8MB (11x9):
- Set the asymmetric DRAM bits for Bank i in SYSCFG 24h as x9.
- Write address 00000000h with 55555555h.
- Write address 00400000h (A22 = 1) with data AAAAAAAAh.
- Write address 00000800h (A11 = 1) with data FEDCBA98h.
- Read from address 00000000h.
  - If it is 55555555h, Bank i has 11x9 DRAM. Store this information, for updating the size registers 13h and 14h after exiting from this program. Increment i and continue from Step 3.
  - If the data that is read back is not 55555555h, continue with Step 9A.

9A. Test of 4MB (11x8):
- Set the size for Bank i as 4MB.
- Also set the asymmetric DRAM bits for Bank i in SYSCFG 24h as x8.
- Write address 00000000h with 55555555h.
- Write address 00200000h (A21 = 1) with data AAAAAAAAh.
- Read from address 00000000h.
  - If it is 55555555h, Bank i has 11x8 DRAM. Store this information, for updating the size registers 13h and 14h after exiting from this program. Increment i and continue from Step 3.
  - If the data that is read back is not 55555555h, go to Step 9B.

9B. Test of 4MB (10x9) and 2MB (10x8 or 9x9):
- Set the size for Bank i as 4MB.
- Also set the asymmetric DRAM bits for Bank i in SYSCFG 24h as “00”.
- Write address 00000000h with 55555555h.
- Write address 00000800h (A11 = 1) with pattern AAAAAAAAh.
- Write address 00200000h (A21 = 1) with data FEDCBA98h.
- Read from address 00000000h.
  - If it is 55555555h, Bank i has 4MB (10x9) DRAM. Store this information, for updating the size registers 13h and 14h after exiting from this program. Increment i, and continue from Step 3.
  - If the data that is read back is AAAAAAAAh, Bank i has 2MB (10x8) DRAM. Store this information, for updating the size registers 13h and 14h after exiting from this program. Set the asymmetric bits corresponding to this bank in SYSCFG 24h for “x8” DRAM. Increment i and continue from Step 3.
  - If the data that is read back is FEDCBA98h, Bank i has 2MB (9x9) DRAM. Store this information, for updating the size registers 13h and 14h after exiting from this program. Increment i and continue from Step 3.

EXIT: Follow the guidelines specified in Table 4-32 for programming DRAM size and asymmetricity for each bank. Set the appropriate bits and exit from DRAM sizing routine.

Note: After a write, and before a read operation, another valid address must be written with data 00000000h to clear bus capacitance.

4.6.6 DRAM Cycles

The fastest possible burst read is 6-2-2-2 which means the first quad-word is received in six clocks and the next three quad-words are received after two clocks each. For a cache based system, it would mean the bursting to the cache and CPU for read miss cycles or write miss cycles. Table 4-35 summarizes the DRAM cycle lengths and the following subsections describe the read/write cycle operations.

4.6.6.1 DRAM Read Cycle

The DRAM read cycle begins with the DRAM controller detecting a page hit or a page miss cycle at the end of the first T2. Based on the status of the current open page and the active RASx#, a page hit, a page miss with RAS inactive, or a page miss with RAS active cycle is executed.

Page Miss with RASx# High Cycle: The row address is generated from the CPU address bus. (Refer back to Table 4-33 and Table 4-34 for the row/column address mux map.) After RASx# goes active, the row address is changed on the next clock edge (programmable to be two CLKs) to the column address. The CASx# will be active two CLKs after the column address is generated.

Page Miss with RASx# Low Cycle: RAS is first precharged for the programmed number of CLKs and then driven active, after which it will be the same as a page miss with RASx# high cycle.

Page Hit Cycle: The SYSC (system controller) generates the column address from the CPU address bus and CASx# is driven active for two clocks. Data flow from the CPU data bus to the memory data bus and vice versa is controlled by the internal DBCOE#, MMDOE#, MDOE#, and HDOE# signals from the SYSC to the DBC. Data from the DRAM is latched by the DBC on the rising edge of each DLE (for CPU reads from DRAM, the DLE[1:0]# signals are identical to the CAS signal). The latched data is valid on the CPU data bus until the next rising edge of CASx#.

During this time, the next read is started, CASx# signals are precharged for one or two clocks (programmable via SYSCFG 02h[1]), and the next data from the DRAM is accessed and latched. The DBC latches the data from the DRAM and holds the data for the CPU while the DRAM controller begins the read for the next word in the burst cycle. The burst read from the DRAM is in effect pipelined into the CPU data bus by FireStar. This scheme reduces the constraints on the board layout so that routing for the CPU data bus, MD data bus, and CASx# signal lines are less critical and performance can be maintained.

Page Hit Cycle (Extended): Wait states can be added if slower DRAMs are used. In this mode, data from the DRAM is latched by the DBC at the end of each CAS cycle similar to the default mode. The only difference between the two modes is that the CAS low time on reads is increased by one T-state. This eases up on the page mode cycle time and CAS access time parameters.

The DRAM read cycle uses a CAS signal that is active for multiples of T-state boundaries rather than half T-state boundaries. This allows additional address decode setup time and MA bus setup time at the start of the cycle, making the fastest burst cycle 7-2-2-2.

Table 4-35 DRAM Cycle Lengths

CPU Bus Speed	Page Hit Leadoff	Page Miss RAS High Leadoff	Page Miss RAS Active Leadoff	CPU Pipeline Reduces Lead-off Cycle by:	Burst Cycle Length	Continued Burst if Pipelined
Read Burst Cycle						
50MHz FPM DRAM	6 clocks	9 clocks	9+precharge	1 clock	X-2-2-2	X-2-2-2
50MHz EDO DRAM	6 clocks	9 clocks	9+precharge	1 clock	X-2-2-2	X-2-2-2
60/66MHz FPM DRAM	6 clocks	9 clocks	9+precharge	1 clock	X-3-3-3	X-3-3-3
60/66MHz EDO DRAM	6 clocks	9 clocks	9+precharge	1 clock	X-2-2-2	X-2-2-2
Write Burst Cycle						
50MHz FPM/EDO DRAM		4-7-3-3	4-(7+pre)-3+3	1 clock	X-3-3-3	X-3-3-3
60/66MHz FPM/EDO DRAM		4-7-3-3	4-(7+pre)-3+3	1 clock	X-3-3-3	X-3-3-3
50/60/66MHz FPM/EDO DRAM Single Write	3	4	4			3

Single writes can be pipelined. Single reads are not pipelined.

4.6.6.2 DRAM Write Cycle

Posted writes to DRAM improves the write cycle timing relative to the CPU and allows FireStar to perform an independent write burst cycle to DRAM without holding the CPU. FireStar maintains a deep data buffer for DRAM writes so that the CPU write cycle is completed without waiting for the external DRAM cycle. For a burst write cycle, the leadoff cycle time is reduced to four clocks even if the cycle is a non-page hit cycle. For a page hit cycle, the burst write can be completed in 4-3-3-3 with posted write enabled. The posted write buffer in the DBC is controlled by the DLE[1:0]# signals from the SYSC. Effectively, the rising edge of these signals will latch the high 32-bit and the low 32-bit new data respectively, from the CPU bus to the posted write buffer.

Single level posted write cycles are employed to achieve a 4-3-3-3 burst at 66MHz. The data from the CPU is latched in the write buffer of the DBC until CAS goes active one T-state after the first T2 (on a page hit). This provides a fast write mechanism and two wait state writes are maintained for the leadoff cycle within a page (even at 66MHz). The CAS pulse width can be extended by one more T-state to ease the timing constraints on the CAS pulse width requirement for speeds above 66MHz.

Table 4-36 DRAM Operation Programming Bits

7	6	5	4	3	2	1	0
SYSCFG 01h			**DRAM Control Register 1**				**Default = 00h**
Row address HOLD after RAS# active: 0 = 2 CPUCLKs 1 = 1 CPUCLK	RAS# active/ inactive when starting a master cycle: 0 = Active (normal page mode) 1 = Inactive	RAS pulse width used during refresh: 00 = 7 CPUCLKs 01 = 6 CPUCLKs 10 = 5 CPUCLKs 11 = 4 CPUCLKs		CAS pulse width during reads: 0 = 3 CPUCLKs 1 = 2 CPUCLKs For 1 CPUCLK width, refer to SYSCFG 1Ch[0].	CAS pulse width during writes: 0 = 3 CPUCLKs 1 = 2 CPUCLKs	RAS precharge time: 00 = 6 CPUCLKs 01 = 5 CPUCLKs 10 = 4 CPUCLKs 11 = 3 CPUCLKs	
SYSCFG 02h			**Cache Control Register 1**				**Default = 00h**
						DRAM posted write: 0 = Disable 1 = Enable	CAS precharge time: 0 = 2 CPUCLKs 1 = 1 CPUCLK
SYSCFG 1Ch			**EDO DRAM Control Register**				**Default = 00h**
							CAS pulse width during DRAM accesses: 0 = CAS pulse width determined by SYSCFG 01h[3] 1 = CAS pulse width is 1 CPUCLK(2)

(2) The width of the pulse is one CPUCLK for read accesses to banks that are populated with EDO DRAMs (selected by bits [7:2]), resulting in X-2-2-2 burst to EDO DRAM at 50/60/66MHz. SYSCFG 14h[7] and PCIDV0 44h[0] must be set in prior to setting this bit. X-2-2-2 burst cycles enabled by this bit apply only during CPU read bursts to EDO DRAM banks that are enabled in SYSCFG 1Ch[7:2].

Table 4-36 DRAM Operation Programming Bits (cont.)

7	6	5	4	3	2	1	0
SYSCFG 0Ch			**DRAM Hole Higher Address**				**Default = 00h**
	Fast BRDY# generation for DRAM write page hits. BRDY# for DRAM writes generated on: 0 = 4th CPUCLK 1 = 3rd CPUCLK						

4.6.6.3 DRAM Refresh Logic

FireStar supports the following types of refresh schemes:

- Normal refresh
- Non-ISA refresh
- Burst refresh

During normal refresh, the CPU bus is put on BOFF# and the DRAM bus is refreshed. This is the default condition at power-up.

In non-ISA refresh, once the REFRESH# signal is generated internally, the DRAM will be refreshed in the background while the CPU is accessing the internal cache. Non-ISA refresh is performed independently of the CPU and does not suffer from the performance restriction of losing processor bandwidth by forcing the CPU into its hold state. Since non-ISA refresh delivers higher system performance, it is recommended over normal refresh. As long as the CPU does not try to access local memory or the ISA bus during a non-ISA refresh cycle, refresh will be transparent to the CPU. The CPU can continue to execute from its internal and secondary caches as well as execute internal instructions during non-ISA refresh without any loss in performance due to refresh arbitration. If a local memory or ISA bus access is required during a non-ISA refresh cycle, wait states will be added to the CPU cycle until the resource becomes available. Non-ISA refresh also separates refreshing of the ISA bus and local DRAM.

In non-ISA refresh mode, the internal REFRESH# signal is not used. FireStar generates an internal refresh input from the system frequency and does a refresh in the background when the DRAM bus is available. Table 4-37 shows the refresh logic associated register bits.

The DRAM controller arbitrates between CPU DRAM accesses and DRAM refresh cycles, while the ISA bus controller arbitrates between CPU accesses to the ISA bus, DMA and ISA refresh. The ISA bus controller asserts the RFSH# and MRD# commands and outputs the refresh address during ISA bus refresh cycles.

FireStar implements refresh cycles to the local DRAM using CAS-before-RAS timing. The CAS-before-RAS refresh uses less power than RAS-only refresh which is important when dealing with large memory arrays. CAS-before-RAS refresh is used for both normal and non-ISA refresh to DRAM memory.

The periodic REFRESH# request signal that occurs every 15µs, originates from the counter/timer of the integrated 82C206. Requests for refresh cycles are generated by two sources: the counter/timer of the integrated 82C206 or 16-bit ISA masters that activate refresh when they have bus ownership. These ISA masters must supply refresh cycles because the refresh controller cannot preempt the bus master to perform the necessary refresh cycles. 16-bit ISA masters that hold the bus longer than 15µs must supply refresh cycles.

Table 4-37 Refresh Logic Register Bits

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">SYSCFG 12h</td><td colspan="4">Refresh Control Register</td><td colspan="2">Default = 00h</td></tr>
<tr><td></td><td></td><td colspan="2">Suspend mode refresh:
00 = From CPUCLK state machine
01 = Self-refresh based on 32KHz only
10 = Normal refresh based on 32KHz only
11 = Reserved</td><td colspan="2">Slow refresh:
Refresh on:
00 = Every REFRESH#/32KHz falling edge
01 = Alternate REFRESH#/32KHz falling edge
10 = One in four REFRESH#/32KHz falling edge
11 = Every REFRESH#/32KHz toggle</td><td></td><td></td></tr>
<tr><td colspan="2">SYSCFG 27h</td><td colspan="4">Miscellaneous Control Register X</td><td colspan="2">Default = 00h</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td colspan="3">Non-ISA refresh counter:
000 = Disable, use external refresh pin
001 = Reserved
010 = Reserved
011 = Reserved
100 = 66MHz external CPU clock
101 = 60MHz external CPU clock
110 = 50MHz external CPU clock
111 = 40MHz external CPU clock</td></tr>
<tr><td colspan="2">SYSCFG 2Eh</td><td colspan="4">UMA Control Register 2</td><td colspan="2">Default = 00h</td></tr>
<tr><td>Allow SDRAM self-refresh in Suspend mode:
0 = Disable (SDRAM engages auto-refresh mode)
1 = Enable (need to enable SDRAM self-refresh if SYSCFG 12h[5:4] = 01 or 10)</td><td>Allow RFSH# signal from IPC to connect to DRAM controller:
0 = Disable
1 = Enable</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td colspan="2">PCIDV1 64h</td><td colspan="4">PCI Master Control Register 1</td><td colspan="2">Default = 10h</td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td colspan="2"></td><td>ISA refresh:
0 = Enable
1 = Disable, to increase PCI master bandwidth</td></tr>
</table>

Table 4-37 Refresh Logic Register Bits (cont.)

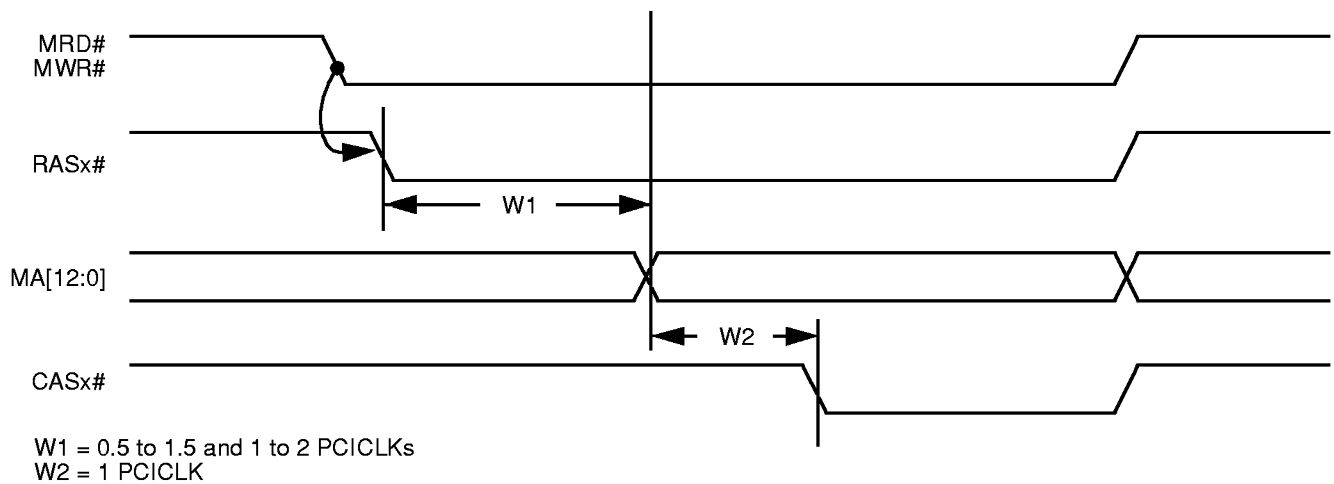
7	6	5	4	3	2	1	0
PCIDV1 64h - FS ACPI Version			**PCI Master Control Register 1**				**Default = 10h**
						Synchronize reset for refresh logic (for improved timing): 0 = Enable 1 = Disable	ISA refresh: 0 = Enable 1 = Disable, to increase PCI master bandwidth

4.6.7 DRAM DMA/Master Cycles

For DMA and master cycles, the DRAM controller operates such that the MRD# and MWR# signals generate RASx# synchronously. The generation of the DRAM column address is then synchronized with PCICLK. The synchronization can be programmed to be 0.5 to 1.5 PCICLKs and 1.0 to 2.0 PCICLKs. The generation of CASx# is always one PCICLK after the generation of the column address. The cycles can thus be completed without adding wait states. For cases when the CPU writeback cache is enabled, wait states need to be added to the DMA/master cycles. This is because the CPU can request a primary cache castout (always a burst write to the DRAMs) and only after the castout is completed can the requested data from the DRAM be fetched.

Note: ISA masters which ignore IOCHRDY may not work when CPU writeback is enabled.

Figure 4-14 ISA Master Synchronization



4.6.8 DRAM Hole Control

FireStar allows system "holes" in DRAM, to which accesses can go to the PCI bus. DRAM holes can be set through SYSCFG 06h[7], 09h[7:0], 0Ah[7:0], 0Bh[7:0], 0Ch[3:0]. Table 4-38 shows these register bits.

Table 4-38 DRAM Hole Control Related Registers

7	6	5	4	3	2	1	0
SYSCFG 06h			**Shadow RAM Control Register 3**				**Default = 00h**
DRAM hole in system memory from 80000h-9FFFFh:(1) 0 = No hole in memory 1 = Enable hole in memory							
(1) This setting gives the user the option to have some other device in the address range 80000h-9FFFFh instead of system memory. When bit 7 is set, the 82C700 will not start the system DRAM controller for accesses to this particular address range.							
SYSCFG 09h			**System Memory Function Register**				**Default = 00h**
DRAM Hole B size: 00 = 512KB 10 = 2MB 01 = 1MB 11 = 4MB Address for this hole is specified in SYSCFG 0Bh[7:0] and 0Ch[3:2]		DRAM Hole B control mode: 00 = Disable 01 = WT for L1 and L2 10 = Non-cacheable for L1 and L2 11 = Enable hole in DRAM		DRAM Hole A size: 00 = 512KB 10 = 2MB 01 = 1MB 11 = 4MB Address for this hole is specified in SYSCFG 0Ah[7:0] and 0Ch[1:0]		DRAM Hole A control mode: 00 = Disable 01 = WT for L1 and L2 10 = Non-cacheable for L1 and L2 11 = Enable hole in DRAM	
SYSCFG 0Ah			**DRAM Hole A Address Decode Register 1**				**Default = 00h**
DRAM Hole A starting address: - These bits along with SYSCFG 0Ch[1:0] are used to specify the starting address of DRAM Hole A. - These bits, AST[7:0], map onto HA[26:19] lines.							
SYSCFG 0Bh			**DRAM Hole B Address Decode Register 2**				**Default = 00h**
DRAM Hole B starting address: - These bits along with SYSCFG 0Ch[3:2] are used to specify the starting address of DRAM Hole B. - These bits, BST[7:0], map onto HA[26:19] lines.							
SYSCFG 0Ch			**DRAM Hole Higher Address**				**Default = 00h**
				DRAM Hole B starting address: These bits are used in conjunction with the bits in SYSCFG 0Bh to specify the starting address of DRAM Hole B. These bits, BST[9:8], map onto HA[28:27].		DRAM Hole A starting address: These bits are used in conjunction with the bits in SYSCFG 0Ah to specify the starting address of DRAM Hole A. These bits, AST[9:8], map onto HA[28:27].	

4.7 CPU Pipelining Control

Depending on the configuration of cache and DRAM, NA# to the CPU must be generated at different times during burst reads or writes. The registers that control NA# generation to the CPU are in summarized in Table 4-39. During CPU bursts, NA# may be generated at three different points. These are controlled by SYSCFG 08h[2], 0Eh[2], 11h[4], and 1Fh[5].

Table 4-39 NA# Generation Control

7	6	5	4	3	2	1	0
SYSCFG 00h		**Byte Merge/Prefetch & Sony Cache Module Control Register**					**Default = 00h**
				Byte/word merging with CPU pipelining (NA# generation) support: 0 = Disable 1 = Enable			
SYSCFG 08h			**CPU Cache Control Register**				**Default = 00h**
					CPU address pipelining for DRAM burst cycles: 0 = Disable 1 = Enable (Allow: X-2-2-2-3-2-2-2 if SYSCFG 1Fh[5] = 1 or X-2-2-2-2-2-2-2 if SYSCFG 1Fh[5] = 0 or X-2-2-2-X-2-2-2 if SYSCFG 1Fh[5] = 0 and 11[4] = 1)		
SYSCFG 0Eh			**PCI Master Burst Control Register 1**				**Default = 00h**
					Generate NA# for every single transfer cycle: 0 = Disable 1 = Enable		

Table 4-39 NA# Generation Control (cont.)

7	6	5	4	3	2	1	0
SYSCFG 0Fh			**PCI Master Burst Control Register 2**				**Default = 00h**
				New mode of single cycle NA#: 0 = No change in cache write hit timing 1 = Cache write hit single transfer cycles will take 3 CLKS to comple if the line is already dirty			
SYSCFG 11h			**Miscellaneous Control Register 2**				**Default = 00h**
			CPU address pipelining for DRAM burst cycles: 0 = Controlled by SYSCFG 08h[2] and 1F[5] 1 = Slow pipelining (allow X-2-2-2-X-2-2-2 when SYSCFG 08h[2] = 1 and 1F[5] = 0				
SYSCFG 17h			**PCI Cycle Control Register 2**				**Default = 00h**
	Generate NA# for PCI slave access in async PCICLK mode: 0 = No 1 = Yes						

Table 4-39 NA# Generation Control (cont.)

7	6	5	4	3	2	1	0
SYSCFG 1Fh			**EDO Timing Control Register**				**Default = 00h**
		NA# generation for burst DRAM accesses: 0 = Aggressive (X-2-2-2-2-2-2-2 if SYSCFG 08h[2] = 1) 1 = Controlled by SYSCFG 08h[2] Also see SYSCFG 11h[4]					
SYSCFG 23h			**Pre-Snoop Control Register**				**Default = 00h**
				Half clock shift of cache hit latching when fast NA is enabled: 0 = Disable 1 = Enable			
SYSCFG 27h			**Miscellaneous Control Register 4**				**Default = 00h**
				Fast NA# with L2 cache: 0 = Disable 1 = Enable			

4.8 PCI Bus Interface

FireStar supports up to four PCI bus masters. Both synchronous and asynchronous modes of operation of the PCI bus, with respect to the CPU, are supported. FireStar supports a 32-bit PCI implementation and supports PCI bus operating frequencies up to 33MHz. It also functions as the PCI-to-ISA expansion bridge and performs the required data path conversion between the 32-bit PCI bus and the 8/16-bit ISA bus.

4.8.1 PCI Master Cycles

A PCI master is always allowed to access the system memory and system I/O spaces. Refer to Table 4-40.

4.8.1.1 System Memory Access

The PCI master asserts FRAME# and puts out the address on the AD[31:0] bus. FireStar decodes that address and provides the data path to the PCI master to access system memory. If the access is to the system memory space, then FireStar acts as the PCI slave and it generates the appropriate control signals to snoop the L1 cache for every access, or for every access to a new line (if the line comparator is enabled).

Table 4-11 and Table 4-12 (on page 59) describe the sequence of events that take place during a master read/write cycle from/to system memory. Listed below is the data flow path for all such accesses by a PCI master.

4.8.1.2 X-1-1-1 Support on PCI Master Cycles

During PCI master read and write burst cycles into DRAM, FireStar has the capability to do X-1-1-1 cycles on the PCI bus. This increases the PCI bandwidth. With the pre-snoop feature of FireStar, a PCI master can sustain bursting to DRAM till a 4K page boundary is reached.

4.8.1.3 Non-Local Memory Access

The PCI master asserts FRAME# and outputs the address on AD[31:0]. If the access is not to the system memory area, FireStar does not assert the internal LMEM#.

All other PCI slaves have up to three PCICLKs after the start of the PCI cycle to assert DEVSEL#. All read/write access from/to PCI slaves is done directly over AD[31:0].

If no PCI slave responds within three PCICLKs after the start of the cycle, then FireStar starts an ISA cycle. For a read access from the ISA bus, the ISA device outputs the data on SD[15:0] or SD[7:0], depending on whether it is a 16- or 8-bit slave. FireStar latches this data and then performs the appropriate data bus conversions and steering (based on the IOCS16#, MEMCS16#, SBHE# signals) and puts the data out on AD[31:0]. For a write access to the ISA bus, the PCI master puts out the data on AD[31:0]. FireStar latches this data and then performs the appropriate data bus conversions and steering (based on the IOCS16#, MEMCS16#, SBHE# signals) and outputs the data on SD[15:0] or SD[7:0], depending on whether it is a 16- or 8-bit slave.

Table 4-40 PCI Master Access Bits

7	6	5	4	3	2	1	0
PCIDV0 04h			**Command Register - Byte 0**				**Default = 07h**
						Memory access (RO): Must = 1 (always) The 82C700 allows a PCI bus master access to memory at anytime. (Default = 1)	I/O access (RO): Must = 1 (always) The 82C700 allows a PCI bus master I/O access at any time. (Default = 1)
PCIDV1 04h			**Command Register - Byte 0**				**Default = 07h**
						Memory access (RO): Must = 1 (always) The 82C700 allows a PCI bus master access to memory at anytime. (Default = 1)	I/O access (RO): Must = 1 (always) The 82C700 allows a PCI bus master I/O access at any time. (Default = 1)

4.8.1.4 PCI Master Pre-Snoop

Pre-snooping is a technique with the aid of which a PCI master can sustain bursting to the local memory till a 4K page boundary is reached. If pre-snooping is enabled, then on the first TRDY# of the PCI master cycle, the state machine within FireStar increments the HA[12:5] address lines by one and asserts EADS# to the CPU after that. By this time, the earlier cache address would have been latched by HACALE. If the CPU responds with a HITM#, then the current PCI master cycle will be terminated at the cache line boundary to allow the writeback cycle to occur. Enabling pre-snooping allows FireStar to continue bursting past a cache line boundary. Table 4-41 shows the register bits associated with the pre-snoop feature.

Table 4-41 Pre-Snoop Control Register Bits

7	6	5	4	3	2	1	0
SYSCFG 0Dh			**Clock Control Register**				**Default = 00h**
						Give FireStar control of the PCI bus on STOP# generation after HITM# is active: 0 = No 1 = Yes(2)	
(2) FireStar has control over the PCI bus until the writeback is completed. If PCI master pre-snoop has been enabled (SYSCFG 0Fh[7] = 1), 0Dh[1] should be set to 1.							
SYSCFG 0Fh			**PCI Master Burst Control Register 2**				**Default = 00h**
PCI pre-snoop: 0 = Disable 1 = Enable(1) <u>Also see SYSCFG 0Dh[1].</u>							
(1) FireStar generates a pre-snoop cycle to the CPU assuming that the PCI master will do a burst.							
SYSCFG 16h			**Dirty/Tag RAM Control Register**				**<u>Default = A0h</u>**
				Pre-snoop control: 0 = Pre-snoop for starting address 0 only 1 = Pre-snoop for all addresses except those on the line boundary			
SYSCFG 1Eh			**Control Register**				**Default = 00h**
		Retry PCI pre-snoop HITM# cycle: 0 = Disable 1 = Enable					

Table 4-41 Pre-Snoop Control Register Bits (cont.)

7	6	5	4	3	2	1	0
SYSCFG 23h			**Pre-Snoop Control Register**				**Default = 00h**
		Pre-snoop for PCI X-1-1-1 write invalidate: 0 = Disable 1 = Enable	Pre-snoop for PCI X-1-1-1 read multiple and read line: 0 = Disable 1 = Enable				

4.8.2 PCI Slave Cycles

4.8.2.1 CPU Master Cycles

Any CPU cycle that is not an access to the system memory area, FireStar translates that cycle to a PCI cycle and asserts FRAME# on the PCI bus. All PCI slaves have up to three PCICLKs after the start of the cycle within which to assert DEVSEL#. The data flow path would be similar to the ones described in the previous section.

4.8.2.2 PCI Byte/Word Merge

This feature, if turned on, allows successive 8-/16-bit writes from the CPU to a PCI slave, to be merged into a 32-bit entity and then sent out to the PCI slave.

To enable the byte/word merge feature, PCIDV1 4Eh[3], PCIDV1 4Eh[1], SYSCFG 17h[2], and SYSCFG 00h[4:3] should be set to 1. Refer to Table 4-42 for information on these register bits.

4.8.2.3 ISA Master Cycles

If the ISA master cycle is not a system memory access, then FireStar becomes the initiator and commences a PCI cycle. The data flow path for an ISA master to a PCI slave access is between the SD[15:0]/SD[7:0] lines and the AD[31:0] lines. FireStar handles all the data bus conversion and steering logic.

Table 4-42 Byte/Word Merge Feature Register Bits

7	6	5	4	3	2	1	0
PCIDV1 4Eh			**Miscellaneous Control Register C - Byte 0**				**Default = 00h**
				Pipelined byte merge function: 0 = Disable 1 = Enable		Byte merge: 0 = Disable 1 = Enable	
SYSCFG 17h			**PCI Cycle Control Register 2**				**Default = 00h**
					Pipelining during byte merge: 0 = Disable 1 = Enable		
SYSCFG 00h			**Byte Merge/Prefetch & Sony Cache Module Control Register**				**Default = 00h**
			Byte/word merge support: 0 = Disable 1 = Enable	Byte/word merging with CPU pipelining (NA# generation) support: 0 = Disable 1 = Enable	Time-out counter for byte/word merge: 00 = 4 CPUCLKs 01 = 8 CPUCLKs 10 = 12 CPUCLKs 11 = 16 CPUCLKs Setting determines maximum time difference between two consecutive PCI byte/word writes to allow merging.		Enable internal HOLD requests to be blocked while performing byte merge: 0 = Disable 1 = Enable

Figure 4-15 CPU 32-Bit Read from PCI

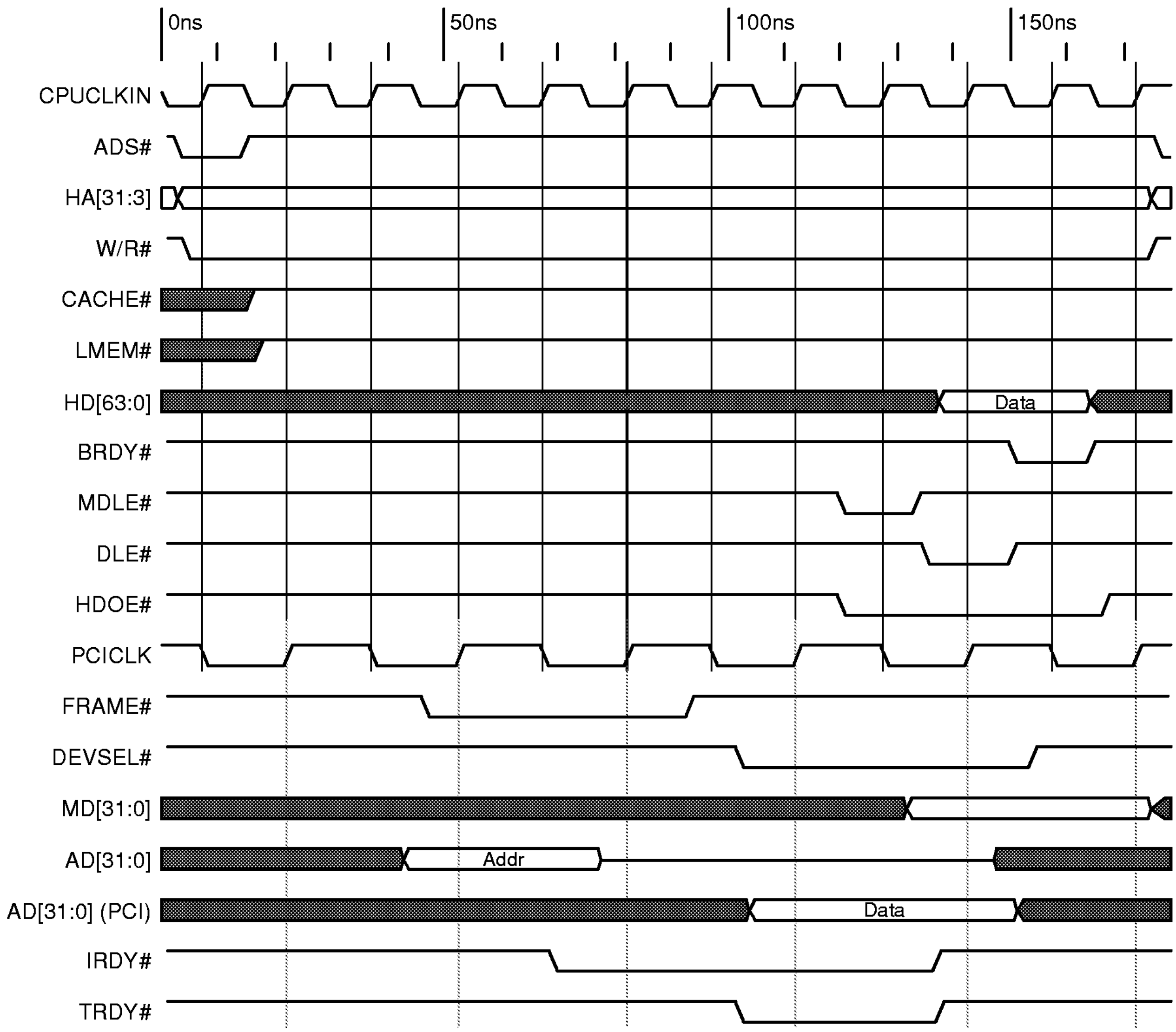


Figure 4-16 CPU 32-Bit Write to PCI

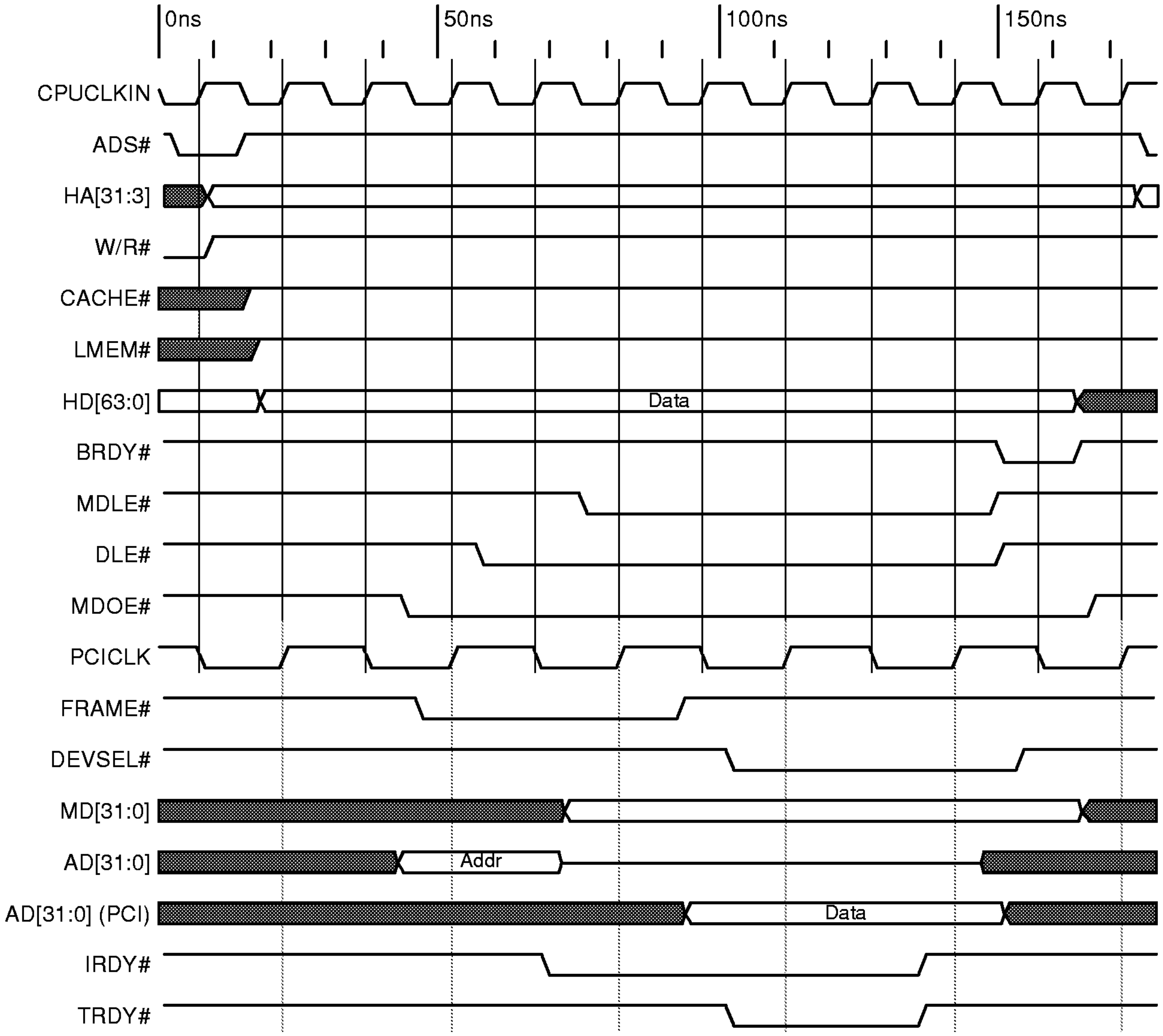


Figure 4-17 CPU 64-Bit Read from PCI

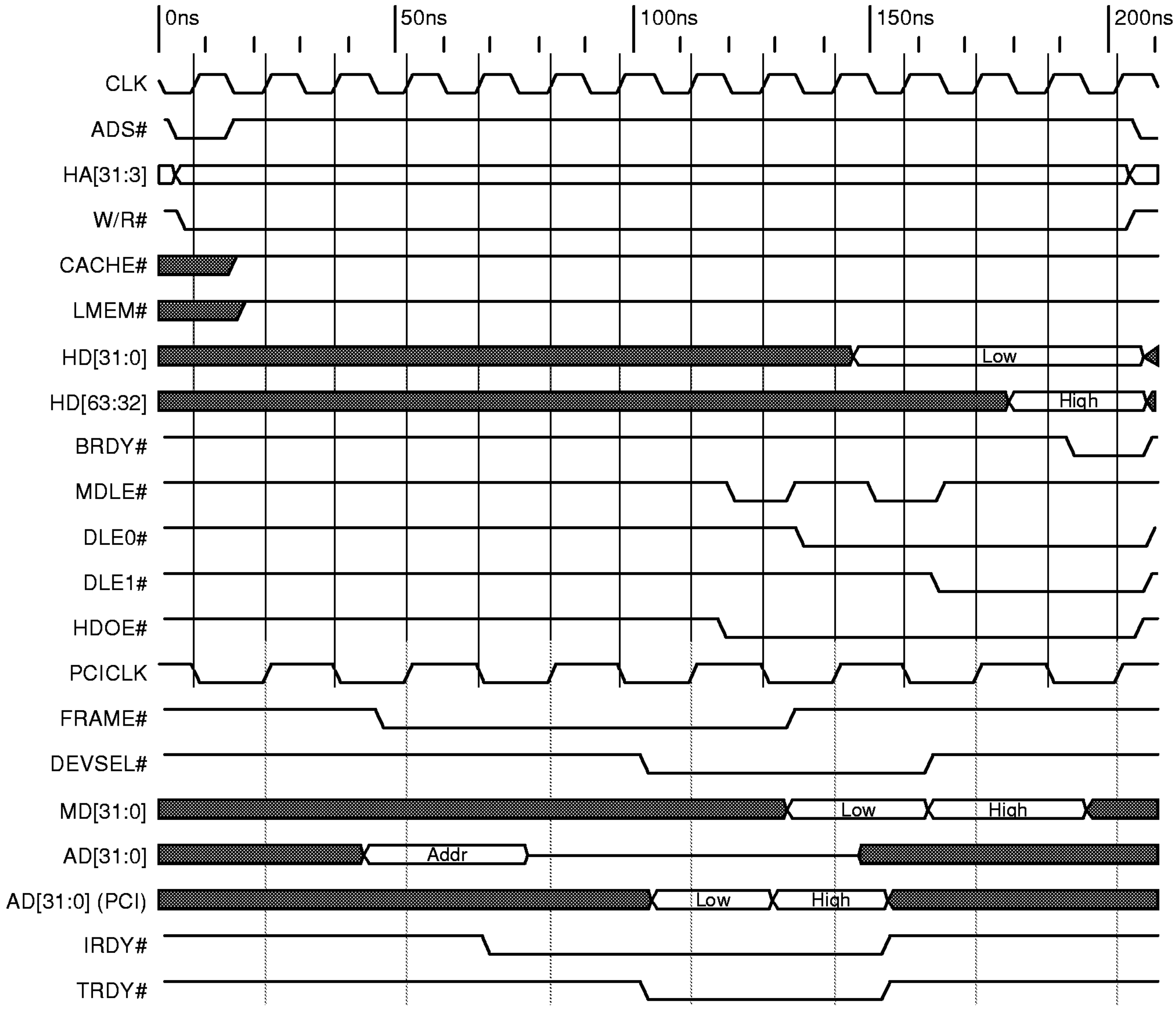


Figure 4-18 ISA Master Read from PCI

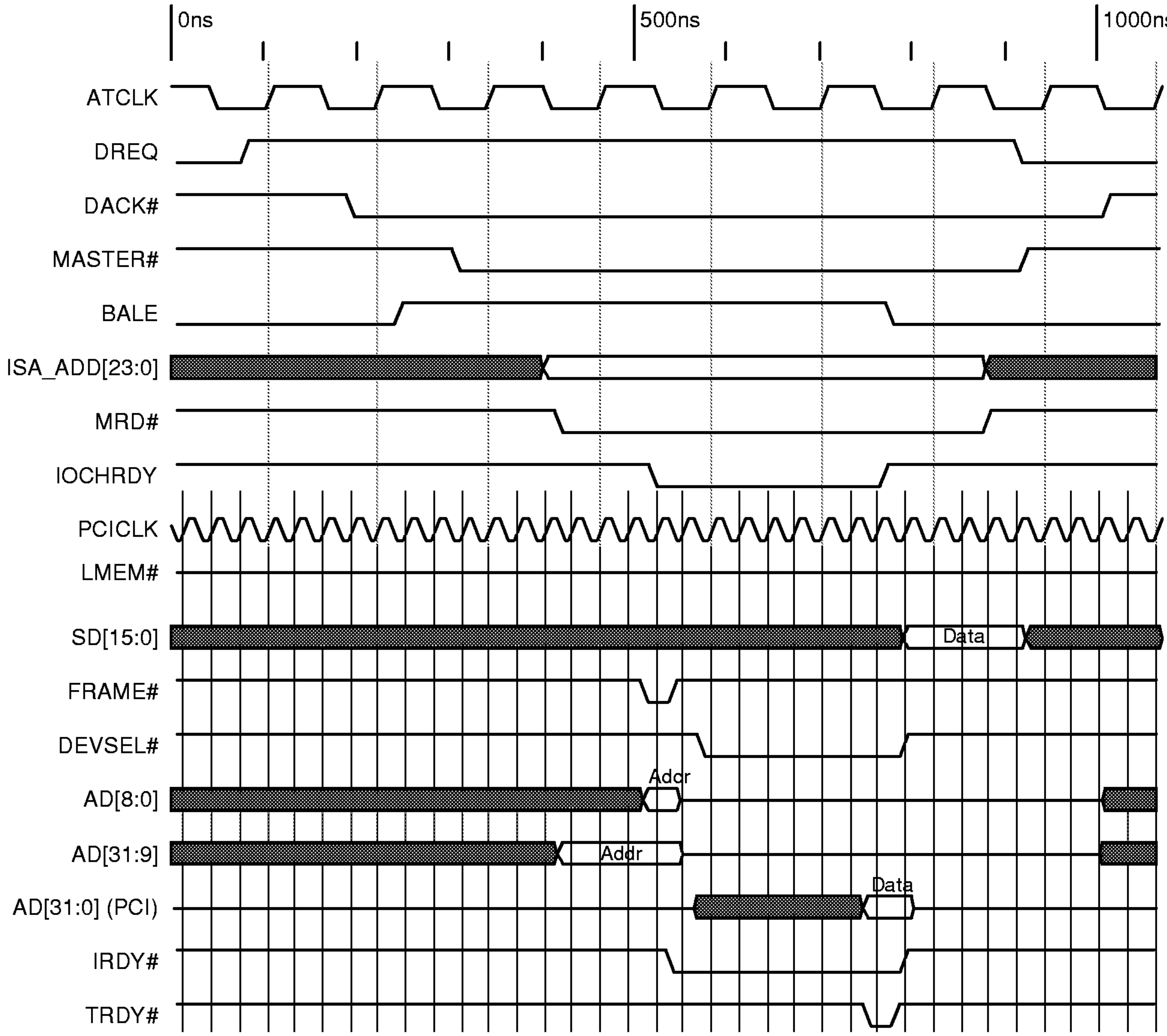


Figure 4-19 ISA Master Write to PCI

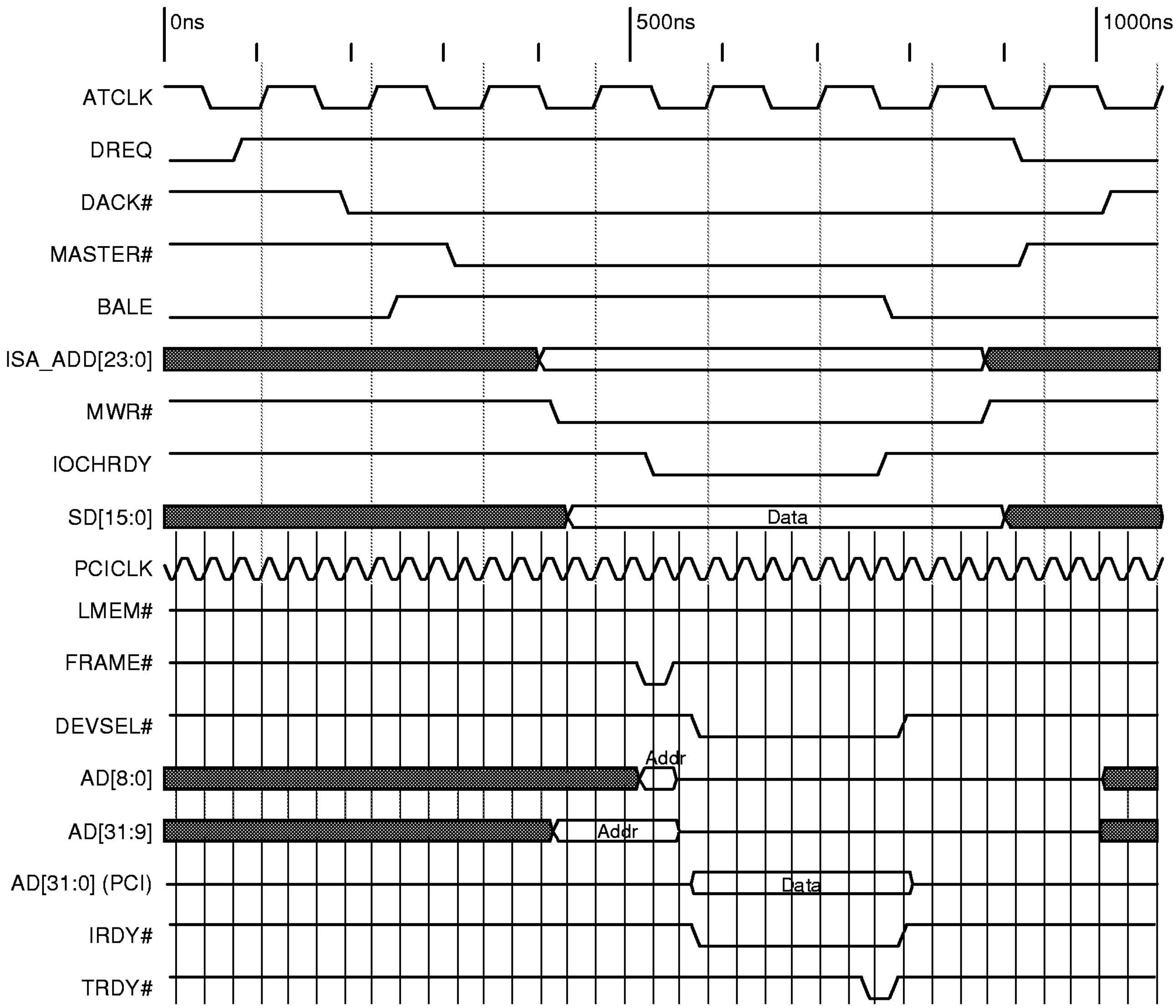


Figure 4-20 ISA Master Read from ISA Slave

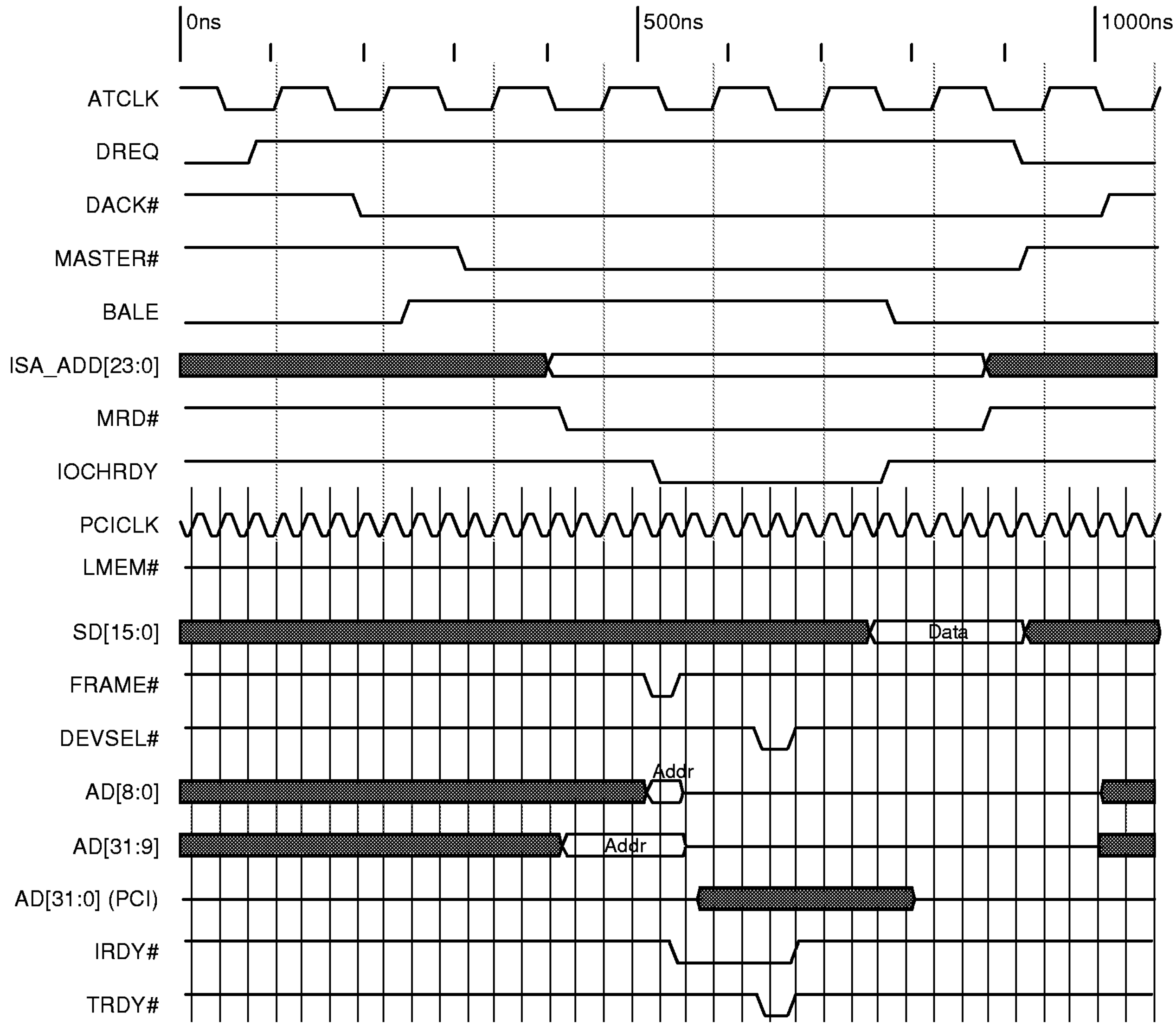
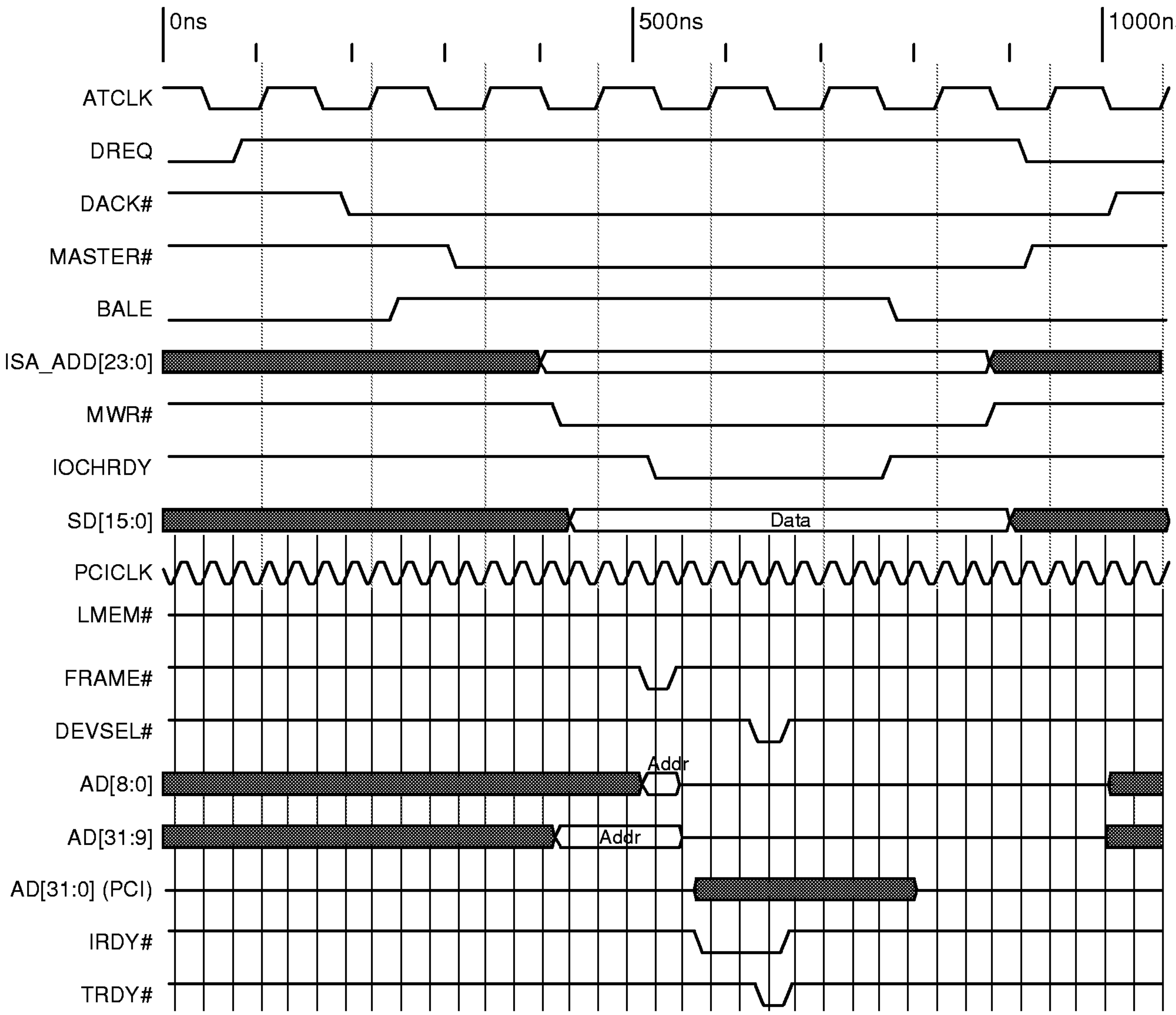


Figure 4-21 ISA Master Write to ISA Slave



4.8.3 PCI Arbitration

The PCI arbiter logic provides up to four external REQ#/GNT# signal pairs. These are utilized as follows.

- REQ0#/GNT0# - always available for PCI bus master devices such as the 82C824 CardBus Controller.
- REQ1#/GNT1# - always available for PCI bus master devices.
- REQ2#/GNT2# - always available for PCI bus master devices.
- REQ3#/GNT3# - available for PCI bus master devices when distributed DMA is not used. If unified memory architecture is adopted, these pins become UFBREQ# and UFBGNT# respectively.

4.9 ISA Bus Interface

The ISA bus state machine gains control when the decoding logic of FireStar detects that no PCI device has claimed the cycle. It monitors status signals M16#, IO16#, IOCHRDY, and NOWS# and performs the necessary synchronization of control and status signals between the ISA bus and the microprocessor. FireStar supports 8/16-bit memory and I/O devices located on the ISA bus.

An ISA bus cycle is initiated by asserting BALE in ISA-TS1 state. On the trailing edge of BALE, M16# is sampled for a memory cycle to determine the bus size. It then enters ISA-TC state and provides the command signal. For an I/O cycle, IO16# is sampled after the trailing edge of ALE until the end of the command. The command cycle is extended when IOCHRDY is detected inactive or the cycle is terminated when the zero wait state request signal (NOWS#) from the ISA bus is active. Upon expiration of the wait states, the ISA state machine terminates itself and passes an internal READY to the CPU state machine to output a synchronous BRDY# to the CPU. The ISA bus state machine also routes data and address when an ISA bus master or DMA controller accesses system memory.

The delay between back-to-back ISA cycles is programmable and can be configured by programming PCIDV1 43h[3:2]. See Table 4-43.

4.9.1 Data Bus Conversion/Data Path Control Logic

Data bus conversion from the 64-bit CPU bus to the memory bus is done by the data buffer controller within FireStar. The data bus conversion from the high order MD bus to the AD bus and the conversion to a 8/16-bit ISA bus is done by FireStar. It converts the CPU byte enables BE[7:0]# to address A2 and four byte enable signals C/BE[3:0]#, for the PCI bus. FireStar uses the C/BE[3:0]#, A2 and the other ISA address (A[1:0], SBHE# and IO16#/M16#) information to complete the 64-bit to 8/16-bit data conversion for the ISA bus. FireStar performs data bus conversion when the CPU accesses 16- or 8-bit devices through 16- or 32-bit instructions. It also handles DMA and ISA master cycles that transfer data between local DRAM or cache memory and locations on the ISA bus.

Table 4-43 Delay Back-to-Back ISA Cycle Register Bit

7	6	5	4	3	2	1	0
PCIDV1 43h			**Feature Control Register**				**Default = 00h**
				Back-to-back ISA I/O cycle delay: 00 = Delay by 3 ATCLKs 01 = Delay by 12 ATCLKs 10 = No delay 11 = Delay by 12 ATCLKs(1)			
(1) When bits [3:2] take on the combination of 11, all back-to-back cycles are delayed by 12 AT clocks. This is different from the combinations of 00 and 01 because in the latter case, the delay will be inserted only when an I/O access is followed by a second I/O access with no other type of access occurring in between (e.g., a memory access).							

4.9.2 Special Cycles

4.9.2.1 System ROM BIOS Cycles

FireStar supports both 8- and 16-bit EPROM cycles. If the system BIOS is 16 bits wide, ROMCS# should be connected to M16# through an open collector gate indicating to FireStar that a 16-bit EPROM is responding. The system BIOS resides on the XD bus.

ROMCS# can generated for both the E0000h-EFFFFh and F0000h-FFFFFh segments through PCIDV1 4Ah and 4Bh. (Refer to Table 4-44.) If a combined video/system ROM BIOS is desired, these two segments should be used.

4.9.2.2 System Shutdown/Halt Cycles

The CPU provides special bus cycles to indicate that certain instructions have been executed or certain conditions have occurred internally. These special cycles, such as shutdown and halt, are covered by dedicated handling logic. FireStar will generate CPUINIT for a CPU shutdown cycle.

Table 4-44 Registers Associated with ROMCS#

7	6	5	4	3	2	1	0
PCIDV1 4Ah			**ROM Chip Select Register 1**				**Default = 00h**
ROMCS# for F8000h-FFFFFh: 0 = Enable 1 = Disable	ROMCS# for F0000h-F7FFFh: 0 = Enable 1 = Disable	ROMCS# for E8000h-EFFFFh: 0 = Disable 1 = Enable	ROMCS# for E0000h-E7FFFh 0 = Disable 1 = Enable	ROMCS# for D8000h-DFFFFh: 0 = Disable 1 = Enable	ROMCS# for D0000h-D7FFFh: 0 = Disable 1 = Enable	ROMCS# for C8000h-CFFFFh: 0 = Disable 1 = Enable	ROMCS# for C0000h-C7FFFh: 0 = Disable 1 = Enable
PCIDV1 4Bh			**ROM Chip Select Register 2**				**Default = 00h**
ROMCS# for FFFF8000h-FFFFFFFFh: 0 = Enable 1 = Disable	ROMCS# for FFFF0000h-FFFF7FFFh: 0 = Enable 1 = Disable	ROMCS# for FFFE8000h-FFFEFFFFh: 0 = Disable 1 = Enable	ROMCS# for FFFE0000h-FFFE7FFFh: 0 = Disable 1 = Enable	ROMCS# for FFFD8000h-FFFDFFFFh: 0 = Disable 1 = Enable	ROMCS# for FFFD0000h-FFFFD7FFFh: 0 = Disable 1 = Enable	ROMCS# for FFFC8000h-FFFCFFFFh: 0 = Disable 1 = Enable	ROMCS# for FFFC0000h-FFFC7FFFh: 0 = Disable 1 = Enable

4.9.3 Local ISA Support Options

The FireStar solution assumes that ISA bus requirements will be minimal in most cases. However, it also makes options available for full ISA bus recovery. The ISA controller configurations are based on the following architecture.

- ISA bus address bits SA[23:0] are generated on dedicated pins.
- Data bits D[15:0] for ISA and Compact ISA cycles are accessed on dedicated pins SD[15:0]. Data bits XD[7:0] for X bus cycles are provided directly on XD[7:0].
- ISA-coupled docking solutions are possible, but are not encouraged. ISA is available on PCI-coupled docking solutions through the 82C825 PCI-ISA Bridge chip.

The configurations possible are described in the following sections.

4.9.3.1 No ISA Bus Configuration

It is possible to use FireStar in a configuration with no local ISA bus. In this case, there is no local X-bus support and all X-bus devices must be positively decoded on PCI by an external device.

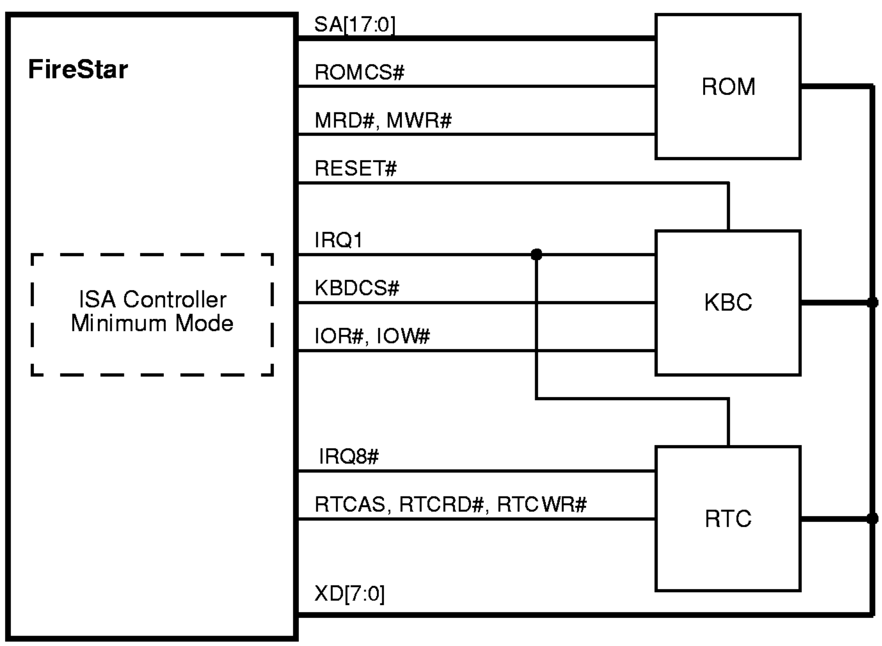
In this configuration, the primary IDE cable device is directly supported (no external buffers needed for isolation) on the SD[15:0] (Cable 0 data) and XD[7:0] buses. Address lines SA[15:0] can then be reassigned as the Cable 1 data bus. Various other pins are switched to directly support the second drive cable.

4.9.3.2 Minimum ISA Bus Configuration

The typical ISA bus utilization in a notebook or ISA-less desktop system requires support for only three devices: the BIOS ROM, the keyboard controller, and the real-time clock (RTC). (Refer to Figure 4-22) FireStar provides direct support for these devices. Devices such as the super I/O chip and audio chip sit directly on PCI in this implementation. The complete set of ISA interface signals would never be required.

Because of the minimal signal set requirements in this mode, many ISA pins are recoverable on the chip and become useful as programmable input/output (PIO) pins as described elsewhere in this document. Typically about 25 pins would become available for other uses.

Figure 4-22 Minimum ISA Bus Configuration

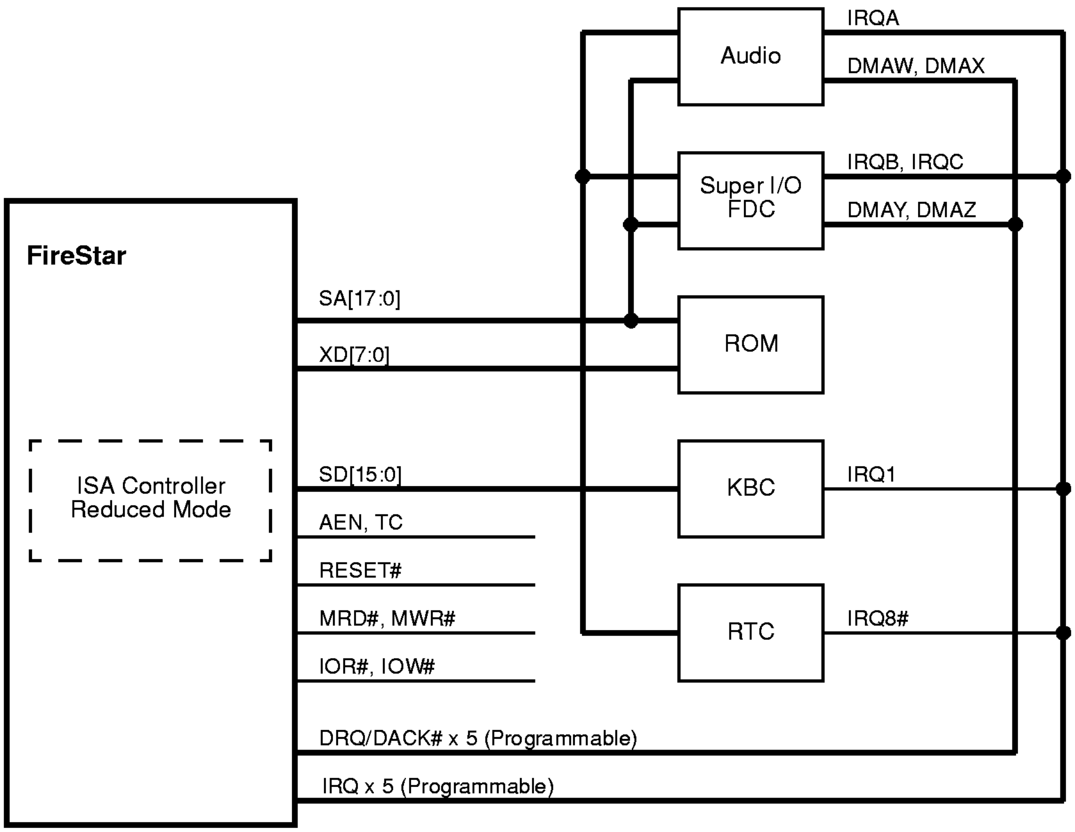


4.9.3.3 Reduced ISA Bus Configuration

Notebook systems that must still rely on ISA devices often require only support for such components as the audio chip and floppy controller. This configuration is much simpler to implement and still leaves about 15 unused ISA pins available for other uses.

Figure 4-23 illustrates the reduced ISA application. Connections to the ROM, KBC and RTC are the same as shown in the previous figure.

Figure 4-23 Reduced ISA Bus Configuration



4.9.3.4 Full ISA Bus Configuration

Full ISA support is possible with no external TTL. The complete 24-bit ISA address is provided. Certain pins, such as MASTER# or IOCHCK#, that are not always useful in a notebook design can be reassigned to other functions such as power control pins through the PIO pin feature. Figure 4-24 shows the configuration for full ISA application.

4.9.3.5 Compact ISA Support

The OPTi Compact ISA (CISA) interface provides full 16-bit support, using only 23 pins, to 100-pin devices such as the OPTi 82C852 PCMCIA Controller. Most signals are shared with existing ISA signals.

Pin Requirements

Compact ISA multiplexes address, IRQ, DRQ, and data all on the ISA SD bus and requires only two dedicated signal pins, CMD# and SEL#/ATB#. Four other signal pins, ATCLK, BALE, IOCHRDY, and RSTDRV, are shared with the standard ISA bus. SPKROUT replaces the SPKD signal on previous OPTi chipsets and allows all devices to combine their speaker outputs with no extra logic.

These minimal pin requirements make Compact ISA very practical for local 16-bit devices that are too slow to need PCI bus speeds.

Figure 4-24 Full ISA Bus Configuration

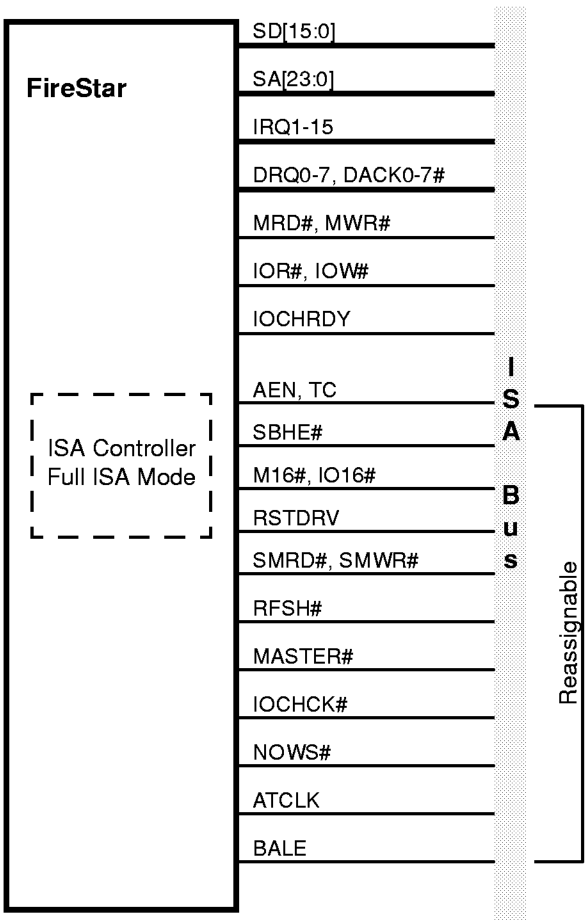
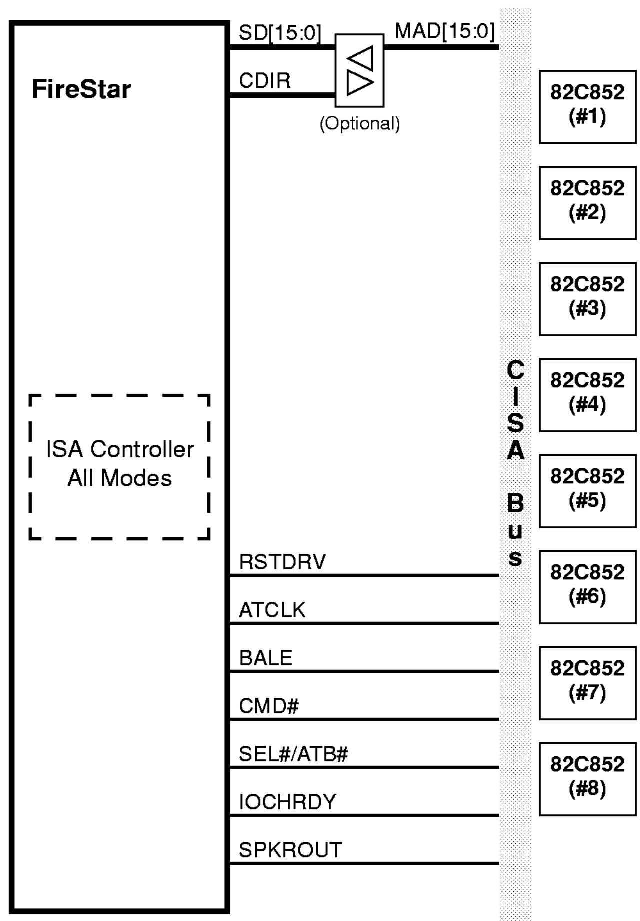


Figure 4-25 Compact ISA Support (available for all configurations)



4.9.3.6 Compact ISA Interface

FireStar incorporates the OPTi Compact ISA (CISA) interface. This interface allows connection of any Compact ISA peripheral device, such as the OPTi 82C852 PCMCIA Controller. The Compact ISA Specification is a separate document (Section Appendix A., "Compact ISA Specification") that describes the interface in detail.

The Compact ISA implementation must deal with certain issues that are specific to the interface architecture.

- ATCLK cannot be stopped without a specific stop clock cycle, since CISA depends on clock edges to transfer interrupts. FireStar can be programmed to generate this stop clock cycle, both automatically and manually.
- The CISA interface generates an AT Backoff (ATB#) signal to FireStar to make an interrupt or DMA request. The CISA interface is required to backoff any ISA cycle it has already started as long as it has not yet asserted ALE. ATB# will come, at latest, one-half ATCLK before ALE# would be asserted. Once ATB# is asserted, FireStar must inhibit all DMA activity and must prevent an EOI command to the interrupt controller from taking effect until ATB# is deasserted and the new DRQ/IRQ states are latched in.
- The FireStar Compact ISA involves two mandatory signals and one optional signal:
  - CMD# (O) - Command, generated by FireStar to run CISA cycles. An external pull-up resistor is required on this line.
  - SEL#/ATB# (I) - CISA peripheral device “selected” handshake input during AT cycles; AT backoff request between cycles; clock restart request during Idle mode. An external pull-up resistor is required on this line.
  - CDIR (O) - Optional CISA buffer direction signal. For desktop-type designs where the CISA signals are buffered on the motherboard to connect through a long ribbon cable to 82C852 PCMCIA Controller(s). CDIR can be programmed on any available PIO pin.

The Compact ISA Control Registers located at SYSCFG F8h, F9h, and FAh enable the interface and control various features.

Table 4-45 Compact ISA Control Registers

7	6	5	4	3	2	1	0
SYSCFG F8h			**Compact ISA Control Register 1**				**Default = 00h**
Inhibit MRD# and MWR# if SEL# asserted on memory cycle: 0 = No 1 = Yes	Inhibit MRD# and MWR# if SEL# asserted on DMA cycle: 0 = No 1 = Yes	Inhibit IOR# and IOW# if SEL# asserted on I/O cycle: 0 = No 1 = Yes	IRQ15 assignment: 0 = IRQ15 1 = RI	Reserved	Fast CISA memory cycle: 0 = Disable (ISA# = 0) 1 = Enable (ISA# = 1)	Reserved	Compact ISA interface: 0 = Disable 1 = Enable If disabled, can use pins as PIO pins.
SYSCFG F9h			**Compact ISA Control Register 2**				**Default = 00h**
SPKD signal driving: 0 = Always, per AT spec 1 = Sync, per CISA spec	End-of-Interrupt Hold - Delays 8259 recognition of EOI command to prevent false interrupts): 00 = None 01 = 1 ATCLK 10 = 2 ATCLKs 11 = 3 ATCLKs		Stop Clock Count bits - Stop clock cycle indication to CISA devices of how many ATCLKs to expect before the clock will stop: 000 = Reserved 001 = 1 ATCLK (Default) ... 111 = 7 ATCLKs			Generate CISA stop clock cycle (if not already stopped): 00 = Never 01 = On STPCLK# cycles to the CPU (hardware) 10 = Immediately (software) 11 = Reserved	
SYSCFG FAh			**Compact ISA Control Register 3**				**Default = 00h**
CDIR response to IDE cable 1 read 0 = Disable 1 = Enable	CDIR response to IDE cable 0 read 0 = Disable 1 = Enable	Reserved		Resume from Suspend on SEL#/ATB# low: 0 = Disable 1 = Enable	Reserved		Configuration cycle generation: 0 = No action 1 = Run cycle using scratchpad

Table 4-45 Compact ISA Control Registers (cont.)

7	6	5	4	3	2	1	0
SYSCFG 6Bh			**Resume Source Register**				**Default = 00h**
DRAM Suspend mode refresh type: 0 = Slow refresh (normal) 1 = Self-refresh	PREQ# caused Resume (RO): 0 = No 1 = Yes	CLKRUN# caused Resume (RO): 0 = No 1 = Yes	Reserved: Write as read.	CISA SEL#/ATB# low caused Resume (RO): 0 = No 1 = Yes	SUSP/RSM caused Resume (RO): 0 = No 1 = Yes	RSMGRP caused Resume (RO): 0 = No 1 = Yes	RI caused Resume (RO): 0 = No 1 = Yes

- **Compact ISA Interface Enable bit (SYSCFG F8h[0]):** Provides master control over whole interface to support CISA. If this bit is 0, all Compact ISA functions are disabled and no address strobing occurs on the SD bus. No other Compact ISA register bits should be set when F8h[0] = 0.
- **IRQ15 Assignment Bit (SYSCFG F8h[4]):** Reassigns IRQ15 from the PCMCIA slot so that it can generate a Ring Indicator (RI) SMI instead.
- **Inhibit Commands if SEL# Asserted (SYSCFG F8h[7:5]):** These bits control whether commands will be hidden from ISA bus peripheral devices if the cycle is claimed by a CISA device. The feature allows devices that use the same memory or I/O space to avoid conflict with each other; CISA devices always preempt ISA devices. A separate bit is provided for memory signals during DMA, which would allow fly-by transfers to function between a PCMCIA DMA card and an ISA memory device.
- **SPKD Signal Driving (SYSCFG F9h[7]):** Selects the CISA scheme for shared audio outputs. Refer to the CISA specification for complete information.
- **CISA Stop Clock Cycle Generation (SYSCFG F9h[1:0]):** enables FireStar to generate the stop clock broadcast cycle on the CISA bus, after which it can stop the AT clock. There are two methods of generating a CISA stop clock cycle: hardware-controlled and software-controlled.
  - Hardware CISA Stop-Clock Control
    - Hardware-controlled CISA stop clock cycle generation occurs automatically, if ATCLK has not been stopped already, whenever SYSCFG F9h[1:0] = 01 and FireStar receives a stop grant cycle from the CPU. FireStar generates a stop request to the CPU when changing CPU speeds or stopping the CPU clock; the CPU responds with a stop grant.
    - When SYSCFG F9h[1:0] = 01 to enable automatic stop clock cycle generation on CISA, address phase 1 of each CISA cycle will not be generated until the cycle is decoded to be an ISA cycle. The logic adds in one extra AT clock before the cycle starts to properly start the CISA interface. Inhibition of CISA phase 1 generation saves power by avoiding unnecessary toggling on the MAD bus.
    - When SYSCFG F9h[1:0] = 00 to disable hardware stop clock mode, FireStar's logic drives address phase 1 of each CISA cycle as soon as it detects ADS# active. In this mode, there is no AT clock start-up delay. Software stop-clock control can still be used to stop the clock and save power.
  - Software CISA Stop-Clock Control
    - Software-controlled CISA stop clock cycle generation occurs only when SYSCFG F9h[1:0] = 10. A CISA stop clock cycle is forced onto the CISA bus. Whenever F9h[1:0] = 10, F9h[7:2] bits written to this register are ignored so no "read/modify/write" procedure is required. This cycle is generated only once; the bits then revert to their previous setting (00 or 01).
- **Stop Clock Count Bits (SYSCFG F9h[4:2]):** Indicate to CISA devices how many ATCLKs to expect before the clock will stop. The default setting of one ATCLK is correct for most applications.
- **CMD# State During Suspend:** If the CISA bus devices are to be powered down during suspend mode, setting PCIDV1 73h[7:6] controls the CMD# line with the same timing as the PPWR0-1 lines so that there is no current leakage path.
- **Resume from Suspend on SEL#/ATB# low:** Setting SYSCFG FAh[3] = 1 allows CISA devices in stop clock mode to resume system operation by generating an interrupt. During normal operation when CISA devices are in stop clock mode, the SEL#/ATB# line acts as a CLKRUN# signal. This bit also allows the same signal to act as RSM#.
- **CISA SEL/ATB# Low Caused Resume** - If SYSCFG FAh[3] = 1 to allow resume from SEL#/ATB#, SYSCFG 6Bh[3] reads 1 to identify the resume source as CISA.

Configuration Cycle Generation

FireStar can be programmed to generate one CISA configuration cycle, the Stop Clock Broadcast cycle, automatically after a period of inactivity. In order to provide for future Configuration Cycle possibilities, the FireStar CISA interface also includes a generic command generation scheme. This scheme takes advantage of the Scratchpad registers already present, and does not prevent their continued use as Scratchpad registers (see Table 4-46). They must be reprogrammed only in order to send out a configuration cycle.

To generate a configuration cycle:

1. Load the phase 1 word in SYSCFG 6Ch-6Dh.
2. Load the phase 2 word in SYSCFG 6Eh-6Fh.
3. Load the data phase word in SYSCFG 52h-53h.
4. Write SYSCFG FAh[0] = 1 to run the cycle.

The CISA interface will generate the desired configuration cycle. The cycle will always be a Broadcast (write) cycle, since there is no inherent means of receiving information back from the configuration cycle. Whenever SYSCFG FAh[0] = 1, the FAh[7:1] bits written to this register are ignored so no “read/modify/write” procedure is required. FAh[0] is automatically cleared to 0 after the cycle runs.

Table 4-46 Scratchpad Registers Used for CISA Configuration Cycles

7	6	5	4	3	2	1	0
SYSCFG 52h			**Scratchpad Register 1**				**Default = 00h**
General purpose storage byte: - For CISA Configuration Cycles: Data phase information, low byte							
SYSCFG 53h			**Scratchpad Register 2**				**Default = 00h**
General purpose storage byte. - For CISA Configuration Cycles: Data phase information, high byte							
SYSCFG 6Ch			**Scratchpad Register 3**				**Default = 00h**
General purpose storage byte - For CISA Configuration Cycles: Address phase 1 information, low byte							
SYSCFG 6Dh			**Scratchpad Register 4**				**Default = 00h**
General purpose storage byte - For CISA Configuration Cycles: Address phase 1 information, high byte							
SYSCFG 6Eh			**Scratchpad Register 5**				**Default = 00h**
General purpose storage byte - For CISA Configuration Cycles: Address phase 2 information, low byte							
SYSCFG 6Fh			**Scratchpad Register 6**				**Default = 00h**
General purpose storage byte - For CISA Configuration Cycles: Address phase 2 information, high byte							
SYSCFG FAh			**Compact ISA Control Register 3**				**Default = 00h**
CDIR response to IDE cable 1 read 0 = Disable 1 = Enable	CDIR response to IDE cable 0 read 0 = Disable 1 = Enable	Reserved		Resume from Suspend on SEL#/ATB# low: 0 = Disable 1 = Enable	Reserved		Configuration cycle generation: 0 = No action 1 = Run cycle using scratchpad

4.9.4 Additional ISA Features

To minimize the performance impact of ISA devices on system performance, FireStar adds several enhancements to the standard ISA subsystem.

4.9.4.1 PCI Positive Decode for ISA

FireStar accommodates the remote ISA bus of the 82C825 ISA Docking Station Bridge through a positive decode of all known device accesses on the local ISA bus. In this way, the only cycles passed through to the docking station PCI bus (for claiming by the secondary ISA bus bridge, the 82C825) are those that more likely than not belong to a docking station device. This method is very practical for a notebook design, since the access ranges of the devices that reside on the local ISA bus are generally all known in advance.

The PMU of the FireStar chip provides power management of on-board devices through ten access event PMIs, of which six are fixed and four are programmable. These same decode ranges are used to determine whether a device on the ISA bus is local and should be claimed without waiting for other PCI devices to respond. Cycles claimed by the chip in this way are always claimed with fast DEVSEL# assertion.

When positive decode occurs, the cycle can either be mapped to a high-order base address to prevent other PCI devices from claiming it, or it can be generated in its normal address space. The SYSCFG registers in Table 4-47 indicate a "positive decode" option and a "positive decode, SMI" option to select the mode desired.

Table 4-47 PCI Positive Decode Ranges for ISA Devices

7	6	5	4	3	2	1	0
SYSCFG AEh			**GNR_ACCESS Feature Register**				**Default = 03h**
GNR set select: 0 = GNR1-4 1 = GNR5-8	Reserved	GNR2 cycle decode type: 0 = I/O 1 = Memory	GNR1 cycle decode type: 0 = I/O 1 = Memory	GNR2 base address: A0 (I/O) A14 (Memory)	GNR1 base address: A0 (I/O) A14 (Memory)	GNR2 mask bit: A0 (I/O) A14 (Memory)	GNR1 mask bit: A0 (I/O) A14 (Memory)
SYSCFG 5Ah if AEh[7] = 0			**PMU Event Register 3**				**Default = 00h**
GNR1_TIMER PMI#11 GNR1_ACCESS PMI#15: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		KBD_TIMER PMI#10 KBD_ACCESS PMI#14: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		DSK_TIMER PMI#9 DSK_ACCESS PMI#13: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		LCD_TIMER PMI#8 LCD_ACCESS PMI#12: 00 = Disable 01 = Reserved 10 = Reserved 11 = SMI	
SYSCFG 5Ah if AEh[7] = 1			**PMU Event Register 3**				**Default = 00h**
GNR5_TIMER PMI: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		Reserved					
SYSCFG D8h if AEh[7] = 0			**PMU Event Register 5**				**Default = 00h**
HDU_TIMER PMI#19 HDU_ACCESS PMI#23: 00 = Disable 01 = Reserved 01 = Reserved 11 = SMI		COM2_TIMER PMI#18 COM2_ACCESS PMI#22: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		COM1_TIMER PMI#17 COM1_ACCESS PMI#21: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		GNR2_TIMER PMI#16 GNR2_ACCESS PMI#20: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI	
SYSCFG D8h if AEh[7] = 1			**PMU Event Register 5**				**Default = 00h**
Reserved						GNR6_TIMER PMI#16 GNR6_ACCESS PMI#20: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI	

7	6	5	4	3	2	1	0
SYSCFG E9h if AEh[7] = 0			**PMU Event Register 7**				**Default = 00h**
GNR4_TIMER PMI#30 GNR4_ACCESS PMI#32: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		GNR3_TIMER PMI#29 GNR3_ACCESS PMI#31: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		GNR4_ ACCESS PMI#32 on current access: 0 = No 1 = Yes	GNR4_ ACCESS PMI#32 on next access: 0 = No 1 = Yes	GNR3_ ACCESS PMI#31 on current access: 0 = No 1 = Yes	GNR3_ ACCESS PMI#31 on next access: 0 = No 1 = Yes
SYSCFG E9h if AEh[7] = 1			**PMU Event Register 7**				**Default = 00h**
GNR8_TIMER PMI#30 GNR8_ACCESS PMI#32: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		GNR7_TIMER PMI#29 GNR7_ACCESS PMI#31: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		GNR8_ ACCESS PMI#32 on current access: 0 = No 1 = Yes	GNR8_ ACCESS PMI#32 on next access: 0 = No 1 = Yes	GNR7_ ACCESS PMI#31 on current access: 0 = No 1 = Yes	GNR7_ ACCESS PMI#31 on next access: 0 = No 1 = Yes
SYSCFG D5h			**X Bus Positive Decode Register**				**Default = 00h**
IPC Registers(1): 00 = Reserved 01 = Positive decode 10 = Reserved 11 = Reserved		RTCRD/WR#, RTCAS I/O Ports 70h-71h: 00 = Reserved 01 = Positive decode 10 = Reserved 11 = Reserved		KBDCS# I/O Ports 60, 64, 62, 66, 92h: 00 = Reserved 01 = Positive decode 10 = Reserved 11 = Reserved		ROMCS# memory segments C000h-F000h: 00 = Reserved 01 = Positive decode 10 = Reserved 11 = Reserved	
(1) I/O 20, 21, A0, A1, 40-43, 00-0F, C0-CF, page registers, high page registers, 22, 24, 23 if Index = 01h)							

4.9.4.2 Remote ISA Support

Another means by which FireStar supports dual ISA buses is to claim unclaimed accesses on the PCI bus and forward them to ISA. The feature works as follows. A cycle is presented on the PCI bus, whether by the CPU or by a local PCI master. The 82C824 CardBus/Docking Controller is programmed to claim these cycles directly and pass them to the 82C825 PCI-ISA Bridge for claiming on the docking ISA bus. The 82C824 also retries these cycles to FireStar until it can respond appropriately, so as not to tie up the local PCI bus waiting for ISA to respond.

If the 82C825 determines that no ISA device is positively claiming the cycle, it will generate a Target Abort to the 82C824. However, the 82C824 chip does not generate Target Abort to FireStar; it simple ignores the subsequent retry attempt by FireStar (or other PCI master). When the 82C824 chip ignores the retry, FireStar claims it on the subtractive decode clock and completes the cycle.

This mechanism must be very well defined, since there is no inherent means for ISA bus devices to claim a cycle. For a generic access, the operation takes place as follows.

1. FireStar remaps the access to the ISA Remap Address range and awaits a response.
2. The 82C824 must be programmed to claim this range, so it will always claim the remapped cycle.
3. The 82C824 then passes the access to the docking station, where the 82C825 claims it and runs an ISA cycle.
4. The 82C825 uses the scheme described in the "Claiming ISA Cycles" below to determine whether the cycle has been "claimed" by an ISA device.
5. The 82C825, 82C824, and FireStar complete their cycles and return any data needed as explained in the "Action After the ISA Cycle" section below.

Claiming ISA Cycles

Because ISA does not provide any acknowledgment that a cycle was claimed by a local device, the 82C825 logic uses a special protocol to determine whether or not to claim the cycle.

Positive ISA Decode: The 82C825 logic waits to determine whether the peripheral device has responded by asserting M16#, IO16#, or NOWS#, or by deasserting IOCHRDY. Any of these signalling events indicates that an ISA device is responding to the cycle and the 82C825 logic can positively claim the cycle without waiting for its completion. These events can be individually masked through 82C825 programming if desired.

Write Cycle: If none of the events mentioned above has been detected during a write cycle, the 82C825 generates a Target Abort to the 82C824 secondary PCI bus. The 82C824 ignores the next retry from the master on the primary PCI bus. FireStar then runs the cycle on the local ISA bus. The write will therefore occur to both ISA buses.

Read Cycle: If none of the events mentioned above has been detected during a read cycle, the 82C825 can still determine whether a device is responding by observing the ISA data bus. If SD[7:0] = FFh, the 82C825 generates a Target Abort and the action proceeds as above for a write cycle. However, if SD[7:0] is any value other than FFh, the 82C825 claims the cycle. Only data bits [7:0] are important for this determination, because M16# and IO16# were not sampled active.

Note that the special case, where FFh is valid read data from a docking station ISA device, is handled correctly in this situation. Once the docking station aborts the cycle, FireStar retries the cycle on the local ISA bus where no device should respond. Since the local SD bus is required to implement pull-up resistors, the correct data value of FFh will still be returned.

Action After the ISA Cycle

Once the ISA cycle is complete on the docking station ISA bus, the 82C825 will either terminate the cycle normally on PCI or will generate Target Abort.

82C825 Normal Termination: Normal termination indicates that an ISA device has responded to the cycle. The 82C825 finishes out the cycle on the secondary PCI bus, and the 82C824 finishes out the retried cycle on the primary PCI bus.

82C825 Target Abort: Target Abort termination indicates that the 82C825 could not conclusively determine that an ISA device accepted the cycle. The 82C824 must in this case ignore the next retry on the primary PCI bus. FireStar claims this cycle and passes it to the local ISA bus.

4.9.5 PC98 Support Features

The NEC PC98 system architecture uses different I/O ports for many peripheral devices in the system. System vendors who accommodate this architecture do so by providing the DMA controller, interrupt controller, and RTC in a separate chip.

FireStar makes provisions for this architecture through a strap-selected option. Refer to Section 3.4, "Strap Selected Options" for information. When enabled, the following changes take place in the FireStar logic.

- FireStar ignores accesses to the I/O address ranges 000-06Fh, 080-0FFh. These cycles will go to the PCI bus and will not be claimed by FireStar until after the subtractive decode clock. If no one else claims the cycle, it gets forwarded to ISA but no FireStar device responds.
- Accesses in the I/O address range 070-07Fh are handled as shown in Table 4-48. No access is provided to the RTC/NMI ports normally available at 070-071h.
- The feature described in the Section 5.1.1, "System Configuration Register Index/Data Programmable", which allows Port 022-024h accesses to be remapped elsewhere, defaults to 01h so that all FireStar SYSCFG registers are accessed at 122-124h instead. This includes the single DMA control data register at 023h index 01h, which gets moved to 123h. BIOS can later overwrite the remap selection to move it to another I/O address if desired.

Table 4-48 I/O Address Range 070-07Fh Accesses

I/O Access to:	Go to this Timer Port:	Comparable to ISA Port:
070h	Ignored	
072h	Timer Channel 0	040h
072h	Ignored	
073h	Timer Channel 1	041h
074h	Ignored	
075h	Timer Channel 2	042h
076h	Ignored	
077h	Timer Control	043h
078h-07Fh	Ignored	

These changes allow FireStar to be used with an external PCI-to-Cbus bridge chip to implement a PC98-compatible system.

The NEC PC98 I/O address space is used shown in Table 4-49. Of these functions, FireStar provides only the Timer.

Table 4-49 PC98 I/O Address Space

I/O Port	Even Address	Odd Address
0-F	Interrupt Controller	DMA Controller
10-1F		DMA Controller
20-2F	Calendar Clock	DMA Bank Register
30-3F	Serial Port	System Port
40-4F	Parallel Printer Port	Key I/F
50-5F	NMI Control	
60-6F	CRTC Text	(Reserved)
70-7F	CRTC	Timer
80-8F	Hard Disk Interface	BRANCH
90-9F	1MB FDD Interface	99-:GPIB Switch,9F:6800 Board
A0-AF	CRTC Graphics	Character Pattern ROM
B0-BF	Communication control adapter-RS-232-C extender	BE:1M/640KB FDD Exchange
C0-CF	C8-:640KB FDD Interface	GPIB
D0-FF	F8-:NDP	
100-17F	Open	Open
180-18F	188-:Sound Board	
190-19F	Open	Open
1A0-1AF	EGC Extended Address	
1B0-1FF	Open	Open

4.9.6 Interrupt Support

FireStar supports a total of six interrupt schemes.

1. *Standard ISA* interrupts can be brought in directly.
2. *PCI* interrupts PCIRQ0#, PCIRQ1#, PCIRQ2#, and PCIRQ3# can be mapped internally to any available IRQ line.
3. *PCI IRQ driveback* cycles can generate any ISA interrupt. The 82C824 uses this scheme to generate interrupts in a parallel format back to FireStar over PCI.
4. *Compact ISA IRQ/DRQ driveback* cycles can generate any ISA interrupt or DMA request. The 82C852 uses this scheme to generate interrupts in a parallel format back to FireStar over ISA.
5. The *Intel Serial IRQ* scheme uses two wires, SIN# and SOUT#, along with the PCICLK to transmit interrupts in a serial format.
6. The *Compaq Serial IRQ* scheme uses a single wire, IRQSER, along with the PCICLK to transmit interrupts in a serial format.

The FireStar support provided for each of these interrupt mechanisms is described next.

4.9.6.1 ISA IRQ Implementation

Pins are provided to support ISA interrupts and PCI interrupts; the pins are assignable (see Table 4-50) to one type of interrupt or to the other (but not both at the same time).

When the pins are assigned as ISA interrupts, they can input either a single ISA interrupt or can mux in four interrupts.

Table 4-50 IRQ Programmable Register Bits

7	6	5	4	3	2	1	0
PCIDV1 B0h			**IRQA Interrupt Selection Register**				**Default = 03h**
Engage pull-down on IRQA? 0 = No 1 = Yes	Reserved		Interrupt source: 0 = ISA (edge) 1 = PCI (level)	Interrupt selection on IRQA pin (Default = IRQ3): 0000 = Disable 0001 = IRQ1 0010 = Rsvd 0011 = IRQ3 0100 = IRQ4 0101 = IRQ5 0110 = IRQ6 0111 = IRQ7 1000 = IRQ8# 1001 = IRQ9 1010 = IRQ10 1011 = IRQ11 1100 = IRQ12 1101 = Rsvd 1110 = IRQ14 1111 = IRQ15			
PCIDV1 B1h			**IRQB Interrupt Selection Register**				**Default = 04h**
Engage pull-down on IRQB? 0 = No 1 = Yes	Reserved		Interrupt source: 0 = ISA (edge) 1 = PCI (level)	Interrupt selection on IRQB pin (Default = IRQ4): 0000 = Disable 0001 = IRQ1 0010 = Rsvd 0011 = IRQ3 0100 = IRQ4 0101 = IRQ5 0110 = IRQ6 0111 = IRQ7 1000 = IRQ8# 1001 = IRQ9 1010 = IRQ10 1011 = IRQ11 1100 = IRQ12 1101 = Rsvd 1110 = IRQ14 1111 = IRQ15			
PCIDV1 B2h			**IRQC Interrupt Selection Register**				**Default = 05h**
Engage pull-down on IRQC? 0 = No 1 = Yes	Reserved		Interrupt source: 0 = ISA (edge) 1 = PCI (level)	Interrupt selection on IRQC pin (Default = IRQ5): 0000 = Disable 0001 = IRQ1 0010 = Rsvd 0011 = IRQ3 0100 = IRQ4 0101 = IRQ5 0110 = IRQ6 0111 = IRQ7 1000 = IRQ8# 1001 = IRQ9 1010 = IRQ10 1011 = IRQ11 1100 = IRQ12 1101 = Rsvd 1110 = IRQ14 1111 = IRQ15			
PCIDV1 B3h			**IRQD Interrupt Selection Register**				**Default = 06h**
Engage pull-down on IRQD? 0 = No 1 = Yes	Reserved		Interrupt source: 0 = ISA (edge) 1 = PCI (level)	Interrupt selection on IRQD pin (Default = IRQ6): 0000 = Disable 0001 = IRQ1 0010 = Rsvd 0011 = IRQ3 0100 = IRQ4 0101 = IRQ5 0110 = IRQ6 0111 = IRQ7 1000 = IRQ8# 1001 = IRQ9 1010 = IRQ10 1011 = IRQ11 1100 = IRQ12 1101 = Rsvd 1110 = IRQ14 1111 = IRQ15			

Table 4-50 IRQ Programmable Register Bits (cont.)

7	6	5	4	3	2	1	0
PCIDV1 B4h			**IRQE Interrupt Selection Register**				**Default = 07h**
Engage pull-down on IRQE? 0 = No 1 = Yes	Reserved		Interrupt source: 0 = ISA (edge) 1 = PCI (level)	Interrupt selection on IRQE pin (Default = IRQ7): 0000 = Disable 0001 = IRQ1 0010 = Rsvd 0011 = IRQ3 0100 = IRQ4 0101 = IRQ5 0110 = IRQ6 0111 = IRQ7 1000 = IRQ8# 1001 = IRQ9 1010 = IRQ10 1011 = IRQ11 1100 = IRQ12 1101 = Rsvd 1110 = IRQ14 1111 = IRQ15			
PCIDV1 B5h			**IRQF Interrupt Selection Register**				**Default = 09h**
Engage pull-down on IRQF? 0 = No 1 = Yes	Reserved		Interrupt source: 0 = ISA (edge) 1 = PCI (level)	Interrupt selection on IRQF pin (Default = IRQ9): 0000 = Disable 0001 = IRQ1 0010 = Rsvd 0011 = IRQ3 0100 = IRQ4 0101 = IRQ5 0110 = IRQ6 0111 = IRQ7 1000 = IRQ8# 1001 = IRQ9 1010 = IRQ10 1011 = IRQ11 1100 = IRQ12 1101 = Rsvd 1110 = IRQ14 1111 = IRQ15			
PCIDV1 B6h			**IRQG Interrupt Selection Register**				**Default = 0Ah**
Engage pull-down on IRQG? 0 = No 1 = Yes	Reserved		Interrupt source: 0 = ISA (edge) 1 = PCI (level)	Interrupt selection on IRQG pin (Default = IRQ10): 0000 = Disable 0001 = IRQ1 0010 = Rsvd 0011 = IRQ3 0100 = IRQ4 0101 = IRQ5 0110 = IRQ6 0111 = IRQ7 1000 = IRQ8# 1001 = IRQ9 1010 = IRQ10 1011 = IRQ11 1100 = IRQ12 1101 = Rsvd 1110 = IRQ14 1111 = IRQ15			
PCIDV1 B7h			**IRQH Interrupt Selection Register**				**Default = 0Bh**
Engage pull-down on IRQH? 0 = No 1 = Yes	Reserved		Interrupt source: 0 = ISA (edge) 1 = PCI (level)	Interrupt selection on IRQH pin (Default = IRQ11): 0000 = Disable 0001 = IRQ1 0010 = Rsvd 0011 = IRQ3 0100 = IRQ4 0101 = IRQ5 0110 = IRQ6 0111 = IRQ7 1000 = IRQ8# 1001 = IRQ9 1010 = IRQ10 1011 = IRQ11 1100 = IRQ12 1101 = Rsvd 1110 = IRQ14 1111 = IRQ15			

4.9.6.2 PCI PCIRQx# Implementation

Any of the IRQA-F pins that are not dedicated to ISA IRQs can be used as PCI PCIRQ0-3# simply by setting the corresponding PCIDV1 B0h-B5h[4] = 1 to switch the input from edge mode to level mode.

If all ISA IRQs are needed along with all PCI interrupts, PCI interrupts are available as PIO options. Each PCIRQx# line then becomes individually mappable to ISA IRQs. In the case where PCIRQx# functions are selected on PIO pins, the registers shown in Table 4-51 select the ISA interrupt to which the PCI interrupt will be routed.

4.9.6.3 PCI IRQ Driveback Implementation

IRQ driveback on the FireStar chip is implemented mainly as it is in the Viper-N+ chipset, and is enabled by writing any non-zero IRQ driveback address to PCIDV1 54h-57h (refer to Table 4-52

The following features have been added to the FireStar implementation.

- IRQ2 generates an SMI#. Note that the sense of IRQ2 is still active high. In this way, devices that use IRQ driveback can generate SMI# simply by routing their normal interrupt to IRQ2 without needing to change the polarity of the interrupt generation logic.
- IRQ13 generates an NMI. This feature allows PCI-to-ISA bridges such as the 82C825 chip to return the CHCK# signal from the ISA bus across the PCI bus. The sense of IRQ13 is active high, like all interrupts generated through IRQ driveback.

PCI Interrupt Sharing across FireBridge (82C814)

FireStar provides registers PCIDV1 B8h and B9h to select the ISA IRQs to which FireBridge PCIRQs will be routed. If this routing is used (required for any FireBridge system), IRQs and PCIRQs from the corresponding FireStar IRQ pin is automatically blocked (even if this pin is set to select PCI interrupts).

The only way to share the same PCI interrupt on both the docking station and the local system is to use a PIO pin to route the PCIRQ line. For example, the Baby AT board allows PCIRQ lines to be routed from either ISA IRQF, G, and H or from PIO pins DRQE, F, and G (jumpers J20 and J21). Only if the PIO pins are used can PCIRQ0# from the docking side be routed to the same IRQ as PCIRQ0# on the FireStar Side.

In FireStar ACPI version, this restriction is removed and is controlled by PCIDV1 4Fh[7] (refer to Table 4-53).

Table 4-51 PCI Interrupt Selection Registers

7	6	5	4	3	2	1	0
PCIDV1 B8h			**PCI Interrupt Selection Register 1**				**Default = 00h**
Interrupt selection on PIO PCIRQ1# input (Default = Disable): 0000 = Disable 0001 = IRQ1 0010 = Rsvd 0011 = IRQ3 0100 = IRQ4 0101 = IRQ5 0110 = IRQ6 0111 = IRQ7 1000 = IRQ8# 1001 = IRQ9 1010 = IRQ10 1011 = IRQ11 1100 = IRQ12 1101 = Rsvd 1110 = IRQ14 1111 = IRQ15				Interrupt selection on PIO PCIRQ0# input (Default = Disable): 0000 = Disable 0001 = IRQ1 0010 = Rsvd 0011 = IRQ3 0100 = IRQ4 0101 = IRQ5 0110 = IRQ6 0111 = IRQ7 1000 = IRQ8# 1001 = IRQ9 1010 = IRQ10 1011 = IRQ11 1100 = IRQ12 1101 = Rsvd 1110 = IRQ14 1111 = IRQ15			
PCIDV1 B9h			**PCI Interrupt Selection Register 2**				**Default = 00h**
Interrupt selection on PIO PCIRQ3# input (Default = Disable): 0000 = Disable 0001 = IRQ1 0010 = Rsvd 0011 = IRQ3 0100 = IRQ4 0101 = IRQ5 0110 = IRQ6 0111 = IRQ7 1000 = IRQ8# 1001 = IRQ9 1010 = IRQ10 1011 = IRQ11 1100 = IRQ12 1101 = Rsvd 1110 = IRQ14 1111 = IRQ15				Interrupt selection on PIO PCIRQ2# input (Default = Disable): 0000 = Disable 0001 = IRQ1 0010 = Rsvd 0011 = IRQ3 0100 = IRQ4 0101 = IRQ5 0110 = IRQ6 0111 = IRQ7 1000 = IRQ8# 1001 = IRQ9 1010 = IRQ10 1011 = IRQ11 1100 = IRQ12 1101 = Rsvd 1110 = IRQ14 1111 = IRQ15			

Table 4-52 IRQ Driveback Address Registers

7	6	5	4	3	2	1	0
PCIDV1 54h			**IRQ Driveback Address Register - Byte 0**				**Default = 00h**
IRQ driveback protocol address bits [7:0]: When an external device logic, such as the 82C824 PC Card Controller or the 82C814 Docking Controller, must generate an interrupt from any source, it follows the IRQ Driveback Protocol and toggles the REQ# line to the 82C700. Once it has the bus, it writes the changed IRQ information to the 32-bit I/O address specified in this register. The 82C700 interrupt controller claims this cycle and latches the new IRQ values. This register defaults to a value of 00h, which disables the IRQ driveback scheme.							
PCIDV1 55h			**IRQ Driveback Address Register - Byte 1**				**Default = 00h**
IRQ driveback protocol address bits [15:8]							
PCIDV1 56h			**IRQ Driveback Address Register - Byte 2**				**Default = 00h**
IRQ driveback protocol address bits [23:16]							
PCIDV1 57h			**IRQ Driveback Address Register - Byte 3**				**Default = 00h**
IRQ driveback protocol address bits [31:24]							

Table 4-53 FireStar ACPI Interrupt Sharing Control Bit

7	6	5	4	3	2	1	0
PCIDV1 4Fh			**Miscellaneous Control Register C - Byte 1**				**Default = 20h**
Reserved							
PCIDV1 4Fh - FS ACPI Version			**Miscellaneous Control Register C - Byte 1**				**Default = 20h**
Allow IRQA, B,...H PCI IRQs (when programed as level IRQs) to be shareable with PIO PCI IRQ pins: 0 = Disable 1 = Enable							

4.9.6.4 Intel Serial IRQ Implementation

FireStar supports the Intel standard of Serial IRQs. This two wire approach is very similar to the one-wire Compaq approach, but permits interrupt sharing between two devices on the line without any possible contention between devices.

Only two control bits are required for the Intel serial IRQ scheme: PCIDV1 BBh[1:0]. Refer to Table 4-54.

Operation

The Intel Serial IRQ protocol requires two pins, the SIN# input and the SOUT# output. Once PCIDV1 BBh[0] is set to 1, IRQ15 automatically becomes SIN# and IRQSER becomes SOUT#. In addition to these pins, the CLKRUN# protocol must be enabled to use Intel Serial IRQs.

The sole function of SOUT# is to initiate a serial interrupt protocol sequence by generating a single low pulse; FireStar will never introduce other IRQs into the frame at the starting end. After the SOUT# pulse has been sent out, the Intel Serial IRQ (ISIRQ) logic will keep sampling the SIN# pin. Once the SIN# data pin is sampled low, the ISIRQ logic enters Start state.

The logic passes through all the SMI and IRQ states to sample the SIN# data pin for the corresponding SMI and IRQ values. All the sampled SMI and IRQ values are passed to the 8259 at the same time that they are sampled, without any delay. When all the SMI and IRQ states have been seen, the ISIRQ logic enters the Stop state.

Once in Stop state, the ISIRQ logic will decide whether to initiate another serial interrupt sequence or not by monitoring the PMU stop PCI clock request (CLKRUN#). If such a PMU request is pending, then the ISIRQ logic will stay in the Stop state until the PMU request is removed. If there is no PMU stop PCI clock request, the ISIRQ logic will initiate another serial interrupt sequence and mask the PMU stop PCI request until it has finished one complete serial interrupt sequence.

If the corresponding bit is set, then any IRR, ISR, or EOI instruction will delay the I/O cycle completion. The IOCHRDY deassertion time is equal to the number of Serial Interrupt devices plus 18 PCI clocks.

Table 4-54 Intel Serial IRQ Control Bits

7	6	5	4	3	2	1	0
PCIDV1 BBh			**Serial IRQ Control Register 2**				**Default = 00h**
						SIRQ delays IRR accesses: 0 = No 1 = Yes	Intel SIRQ (Intel serial IRQ scheme): 0 = Disable 1 = Enable

4.9.6.5 Compaq Serial IRQ Implementation

The FireStar chipset supports the Compaq standard of Serial IRQs. This one wire approach is very compact compared to the Intel two-wire approach, but if two devices on the line want to share the same interrupt, there may be brief contention since both devices drive the line low on one clock and high on the clock that immediately follows. Because of this contention, OPTi cannot guarantee against chip hardware failure if interrupts are shared in this mode.

Operation

The Compaq Serial IRQ protocol requires one additional PCI sustained tristate pin, the IRQSER signal. For detailed Serial IRQ operation, refer to the "Serialized IRQ for PCI Systems" specification version 5.3.

After setting PCIDV1 BAh[0] = 1 to enable Compaq Serial IRQ (CSIRQ) mode, the CSIRQ controller initiates a Continuous mode Start frame. During the Data frame, the CSIRQ logic samples the IRQSER input for the corresponding SMI, IOCHCK#, and IRQ values, and then passes the sampled values to 8259.

At the end of the Data frame, the CSIRQ controller will sample the QUIET and HALT bits to determine whether the next Compaq Serial IRQ cycle will be Continuous mode, Quiet mode, or a temporary Halt state.

If the next cycle is sampled to be Continuous mode, IRQSER is asserted for three PCI clocks. Once the logic enters Idle state, it checks whether the PMU stop PCI clock request is pending. If so, the CSIRQ logic will stay in the Idle state until the PMU request is removed.

If the next cycle is sampled to be Quiet mode, IRQSER is asserted for two PCI clocks. Once the logic enters Idle state, it samples the IRQSER input to begin the Quiet mode cycle. Since FireStar has no control of the Start frame, this mode is not recommended for mobile application.

If the HALT bit is sampled active, then the CSIRQ logic asserts IRQSER for three PCI clocks to tell all the Serial IRQ devices that next cycle will be Continuous mode; the logic then enters Halt state. In Halt state, CSIRQ configuration can be changed. Clearing the HALT bit will immediately cause a Continuous mode Start frame to be generated.

If the corresponding bit is set, then any IRR, ISR, and EOI instructions will delay I/O cycle completion. The IOCHRDY de-assertion time is equal to the Serial IRQ cycle time in PCI clocks.

Since the Compaq Serial IRQ specification does not specify the behavior when PCI clock is stopped, FireStar implementation has considered this issue with both CLKRUN# and no CLKRUN# support. In both cases, the Serial IRQ controller always enters into the Idle state before the PCI clock is stopped. Once the PCI clock is restarted, a Continuous mode cycle will be initiated to collect new IRQ values from devices.

If new IRQ information needs to be sent to the FireStar during clock stop mode, the device can do so by asserting CLKRUN# to wake up the PCI clock. However, devices without CLKRUN# support have to wait until the PCI clock is restarted by some other means.

Table 4-55 Compaq Serial IRQ Control Bits

7	6	5	4	3	2	1	0
PCIDV1 BAh			**Serial IRQ Control Register 1**				**Default = 00h**
Compaq SIRQ HALT mode request: 0 = Active 1 = Halt	Compaq SIRQ QUIET mode request: 0 = Continuous 1 = Quiet		Compaq SIRQ data frame slots. Change only when the serial IRQ logic is disabled or in HALT state: 0 = 17 slots 1 = 21 slots	Compaq SIRQ - Start frame width in PCI clocks. Change this setting only when serial IRQ is disabled or in HALT state: 00 = 4 PCICLKs 01 = 6 PCICLKs 10 = 8 PCICLKs 11 = Reserved		SIRQ delays EOI accesses: 0 = No 1 = Yes	Compaq SIRQ (Compaq serial IRQ scheme): 0 = Disable 1 = Enable
PCIDV1 BBh			**Serial IRQ Control Register 2**				**Default = 00h**
Compaq SIRQ in HALT state (RO): 0 = No 1 = Yes	Compaq SIRQ in QUIET state (RO): 0 = No 1 = Yes						

Note: QUIET - PCIDV1 BAh[6] requests the next Serial IRQ cycle to be Continuous or Quiet mode. In mobile applications, use Continuous mode only. This is to guarantee that the host gains control of the Serial IRQ for Suspend and APM stop clock. In applications where the PCI clock never stops, use either mode. PCIDV1 BBh[6] can be read to determine the current state of the logic.

HALT - PCIDV1 BAh[7] requests a temporary halt of the Serial IRQ controller as soon as the current cycle has returned to Idle state. Once in Halt state, the Serial IRQ configuration can be changed. After the logic has been put in Halt state, upon clearing this bit the logic will return to Continuous mode. PCIDV1 BBh[7] can be read to determine the current state of the logic.

4.10 Internal Integrated Peripherals Controller

The following subsections give detailed operational information about the FireStar internal integrated peripheral controller (IPC) which includes two 8237 DMA controllers, two 8259 interrupt controllers, one 8254 timer/counter and one 74612 memory mapper. It is register-compatible with the 82C206 chip.

For information on the design architecture of this unit, refer to the separate document on the 82C206 IPC. This document is available on request from OPTi.

4.10.1 IPC Configuration Programming

The sole configuration register (see Table 4-56) of the internal IPC, separate from those of the 82C700, is accessed by first writing the register index of interest to I/O Port 022h; the selected register information then becomes available for reading or writing at I/O Port 023h as opposed to Port 024h used by FireStar configuration registers.

Table 4-56 Internal IPC Configuration Bits.

7	6	5	4	3	2	1	0
Index 01h: IPC Configuration Register							
IPC register access wait states (ATCLKs): 00 = 1 wait states 01 = 2 wait states 10 = 3 wait states 11 = 4 wait states (Default)		16-bit DMA wait states:(1) 00 = 1 wait state (Default) 01 = 2 wait states 10 = 3 wait states 11 = 4 wait states		8-bit DMA wait states:(1) 00 = 1 wait state (Default) 01 = 2 wait states 10 = 3 wait states 11 = 4 wait states		Delay DMA MEMR# 1 CLK from system MEMR#: 0 = Yes (AT-compatible, Default) 1 = No	DMA clock select: 0 = ATCLK/2, (Default) 1 = ATCLK
(1) IOCHRDY can also be asserted by DMA devices to add wait states to DMA cycles.							

4.10.2 IPC Register Programming

The IPC provides two peripheral interrupt controllers that are register compatible with the 8259 part. The registers of this logic module are listed below. These registers are accessed directly through the I/O subsystem (no index/data method is used).

4.10.2.1 Initialization Command Words

The Initialization Command Words (ICWs) are shown first and must always be written in sequence starting with ICW1. Two I/O port groups are listed. The first group refers to INTC1, the interrupt controller for IRQ[7:0] (see Table 4-57); the second refers to INTC2, the interrupt controller for IRQ[15:8] (see Table 4-58).

Table 4-57 INTC1 Initialization Command Words

7	6	5	4	3	2	1	0
Port 020h				**ICW1 (WO)**			
Don't care			Always = 1	Trigger mode: 0 = Edge 1 = Level	Don't care	Cascade mode select: 0 = Yes (always) 1 = No	Don't care
Port 021h				**ICW2 (WO)**			
V[7:3] - Upper bits of interrupt vector. For AT compatibility, write 08h.					Not used - lower bits of interrupt vector are generated by interrupt controller.		
Port 021h				**ICW3 (WO)**			
- S[7:0] - Slave mode controller connections. For AT compatibility, write 04h (IRQ2).							
Port 021h				**ICW4 (WO)**			
Don't care			Enable multiple interrupts: 0 = No 1 = Yes	Don't care		Enable auto EOI command: 0 = No 1 = Yes	Don't care

Table 4-58 INTC2 Initialization Command Words

7	6	5	4	3	2	1	0
Port 0A0h				**ICW1 (WO)**			
Don't care			Always = 1	Trigger mode: 0 = Edge 1 = Level	Don't care	Cascade mode select: 0 = Yes (always) 1 = No	Don't care
Port 0A1h				**ICW2 (WO)**			
V[7:3] - Upper bits of interrupt vector. For AT compatibility, write 70h.					Not used - lower bits of interrupt vector are generated by interrupt controller.		
Port 0A1h				**ICW3 (WO)**			
Don't care					ID[2:0] - Slave mode address. For AT compatibility, write 02h (IRQ2).		
Port 0A1h				**ICW4 (WO)**			
Don't care			Enable multiple interrupts: 0 = No 1 = Yes	Don't care		Enable auto EOI command: 0 = No 1 = Yes	Don't care

Enable Multiple Interrupts can be enabled to allow INTC2 to fully nest interrupts without being blocked by INTC1. Correct handling of this mode requires the CPU to issue a non specific EOI command to zero when exiting an interrupt service routine. If the feature is disabled, no command need be issued.

Automatic End-of-Interrupt can be enabled to allow the interrupt controller to generate a non specific EOI command on the trailing edge of the second interrupt acknowledge cycle from the CPU. The feature allows the interrupt currently in service to be cleared automatically on exit from the service routine. This function should not be used with fully nested interrupts except by INTC1.

4.10.2.2 Operational Command Words

The Operational Command Words are used to program the interrupt controller during the course of normal operation. Two I/O port addresses are listed for each register. The first address refers to INTC1, the interrupt controller for IRQ[7:0]; the second refers to INTC2, the interrupt controller for IRQ[15:8].

Table 4-59 INTC1 and INTC2 Operational Command Words

7	6	5	4	3	2	1	0
Port 021h, 0A1h	**OCW1 Mask Register**						
IRQ7/15: 0 = Enable 1 = Mask	IRQ6/14: 0 = Enable 1 = Mask	IRQ5/13: 0 = Enable 1 = Mask	IRQ4/12: 0 = Enable 1 = Mask	IRQ3/11: 0 = Enable 1 = Mask	IRQ2/10: 0 = Enable 1 = Mask	IRQ1/9: 0 = Enable 1 = Mask	IRQ0/8: 0 = Enable 1 = Mask
Port 020h, 0A0h	**OCW2 Command Register (WO)**						
000 = Disable auto-rotate, auto EOI mode 100 = Enable auto-rotate, auto EOI mode 001 = Generate non-specific EOI 011 = Generate specific EOI 101 = Rotate on non-specific EOI 111 = Rotate on specific EOI 110 = Set priority 010 = No operation			Always = 0 for OCW2	Always = 0 for OCW2	L[2:0]: Interrupt level acted on by “set priority” and “rotate of specific EOI”		
Port 020h, 0A0h	**OCW3 Command Register (WO)**						
Always = 0	Allow bit 5 changes: 0 = No 1 = Yes	Special mask mode: 0 = Disable 1 = Enable	Always = 0 for OCW3	Always = 1 for OCW3	Polled mode: 0 = Disable (generate interrupt) 1 = Enable (poll 020/0A0h for interrupt)	Allow bit 0 changes: 0 = No 1 = Yes	In-service access: 0 = 020/0A0h reads return IRR 1 = Return ISR
Port 020h, 0A0h	**Interrupt Request Register OCW3[0] = 0 (RO)**						
IRQ7/15 pending: 0 = No 1 = Yes	IRQ6/14 pending: 0 = No 1 = Yes	IRQ5/13 pending: 0 = No 1 = Yes	IRQ4/12 pending: 0 = No 1 = Yes	IRQ3/11 pending: 0 = No 1 = Yes	IRQ2/10 pending: 0 = No 1 = Yes	IRQ1/9 pending: 0 = No 1 = Yes	IRQ0/8 pending: 0 = No 1 = Yes
Port 020h, 0A0h	**In-Service Register OCW3[0] = 1 (RO)**						
IRQ7/15 in service: 0 = No 1 = Yes	IRQ6/14 in service: 0 = No 1 = Yes	IRQ5/13 in service: 0 = No 1 = Yes	IRQ4/12 in service: 0 = No 1 = Yes	IRQ3/11 in service: 0 = No 1 = Yes	IRQ2/10 in service: 0 = No 1 = Yes	IRQ1/9 in service: 0 = No 1 = Yes	IRQ0/8 in service; 0 = No 1 = Yes
Port 020h, 0A0h	**Polled Mode Register OCW3[2] = 1 (RO)**						
Interrupt pending: 0 = No 1 = Yes	Not used				IRQ[2:0]: Number of highest priority interrupt that is pending		

4.10.2.3 Interrupt Controller Shadow Registers

Values written to the interrupt controller are not always directly readable in the AT architecture. However, FireStar shadows these values as they are written so that they can be read back later through the configuration registers. Table 4-60 lists the correspondence of shadow indexes to the write-only registers in the interrupt controllers.

4.10.3 DMA Controller Programming Registers

The integrated IPC provides two direct memory access controllers (DMAC1 and DMAC2) and their associated memory mappers that are register compatible with AT-type systems. The registers of this logic module are listed below. These registers are accessed directly through the I/O subsystem (no index/data method is used). Each DMAC has four DMA channels. Channels 3 through 0 are in DMAC1 and Channels 7 through 4 are in DMAC2. Table 4-61 and Table 4-62 list the register locations.

Table 4-60 Interrupt Controller Shadow Register Index Values

Register	INTC1 Index	INTC2 Index
ICW1	80h	88h
ICW2	81h	89h
ICW3	82h	8Ah
ICW4	83h	8Bh
OCW2	85h	8Dh
OCW3	86h	8Eh

Table 4-61 DMA Address and Count Registers

	DMA Channel Address							
Name	0	1	2	3	4	5	6	7
Memory Address Register	000h R/W	002h R/W	004h R/W	006h R/W	0C0h R/W	0C4h R/W	0C8h R/W	0CCh R/W
Count Register	001h R/W	003h R/W	005h R/W	007h R/W	0C2h R/W	0C6h R/W	0CAh R/W	0CEh R/W
EISA High Byte Count Register	401h R/W	403h R/W	405h R/W	407h R/W	None	4C6h R/W	4CAh R/W	4CEh R/W
Page Address Register	087h R/W	083h R/W	081h R/W	082h R/W	08Fh R/W	08Bh R/W	089h R/W	08Ah R/W
EISA High Byte Page Address Register	487h R/W	483h R/W	481h R/W	482h R/W	None	48Bh R/W	489h R/W	48Ah R/W

Table 4-62 DMA Control and Status Registers

		Command Port Address	
Command	Function	DMA Channels 7-5	DMA Channels 3-0
Mode Register	Sets the function type for each channel. Group can be read back - see “Reset Mode Register Readback Counter” command.	Read/Write 0D6h	Read/Write 00Bh
Status Register	Returns channel request and terminal count information.	Read 0D0h	Read 008h
Command Register	Sets the DMAC configuration.	Write 0D0h, Read 0D4h	Write 008h, Read 00Ah
Request Register	Makes a software DMA request.	Read/Write 0D2h	Read/Write 009h
Mask Register	Enables or masks DMA transfers on selected channels.	Read/Write 0DEh	Read/Write 00Fh
Temporary Register	Not used in AT-compatible design.	Read 0DAh	Read 00Dh

Table 4-63 DMAC1 Control and Status Bits

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td>Port 008h</td><td colspan="7">DMAC1 Status Register</td></tr>
<tr><td>Ch. 3 request pending:
0 = No
1 = Yes</td><td>Ch. 2 request pending:
0 = No
1 = Yes</td><td>Ch. 1 request pending:
0 = No
1 = Yes</td><td>Ch. 0 request pending:
0 = No
1 = Yes</td><td>Ch. 3 reached terminal count:
0 = No
1 = Yes</td><td>Ch. 2 reached terminal count:
0 = No
1 = Yes</td><td>Ch. 1 reached terminal count:
0 = No
1 = Yes</td><td>Ch. 0 reached terminal count:
0 = No
1 = Yes</td></tr>
<tr><td>Port 00Bh</td><td colspan="7">DMAC1 Mode Register</td></tr>
<tr><td colspan="2">Mode select:
00 = Demand 10 = Block
01 = Single 11 = Cascade</td><td>Address count:
0 = Increment
1 = Decrement</td><td>Auto-initialize:
0 = Disable
1 = Enable</td><td colspan="2">Transfer select:
00 = Verify 10 = Mem. read
01 = Mem. write 11 = Reserved</td><td colspan="2">Channel select:
00 = Ch. 0 10 = Ch. 2
01 = Ch. 1 11 = Ch. 3</td></tr>
<tr><td>Port 009h</td><td colspan="7">DMAC1, DMA Request Register</td></tr>
<tr><td colspan="5">Reserved: Write as 0.</td><td>Request:
0 = Clear
1 = Set</td><td colspan="2">Channel select:
00 = Ch. 0 10 = Ch. 2
01 = Ch. 1 11 = Ch. 3</td></tr>
<tr><td>Port 008h</td><td colspan="7">DMAC1 Command Register</td></tr>
<tr><td>DACK active sense:
0 = Low
1 = High</td><td>DRQ active sense:
0 = High
1 = Low</td><td>Extended write:
0 = Disable
1 = Enable</td><td>Rotating priority:
0 = Disable
1 = Enable</td><td>Compressed timing:
0 = Disable
1 = Enable</td><td>DMAC operation:
0 = Enable
1 = Disable</td><td>Channel 0 address hold:
0 = Disable
1 = Enable</td><td>Memory-to-memory:
0 = Disable
1 = Enable</td></tr>
<tr><td>Port 00Fh</td><td colspan="7">DMAC1 Mask Register</td></tr>
<tr><td colspan="4">Reserved: Write as 0.</td><td>Channel 3:
0 = Unmasked
1 = Masked</td><td>Channel 2:
0 = Unmasked
1 = Masked</td><td>Channel 1:
0 = Unmasked
1 = Masked</td><td>Channel 0:
0 = Unmasked
1 = Masked</td></tr>
</table>

Table 4-64 DMAC2 Control and Status Bits

7	6	5	4	3	2	1	0
Port 0D0h	**DMAC2 Status Register**						
Ch. 7 request pending: 0 = No 1 = Yes	Ch. 6 request pending: 0 = No 1 = Yes	Ch. 5 request pending: 0 = No 1 = Yes	Ch. 4 request pending: 0 = No 1 = Yes	Ch. 7 reached terminal count: 0 = No 1 = Yes	Ch. 6 reached terminal count: 0 = No 1 = Yes	Ch. 5 reached terminal count: 0 = No 1 = Yes	Ch. 4 reached terminal count: 0 = No 1 = Yes
Port 0D6h	**DMAC2 Mode Register**						
Mode select: 00 = Demand 01 = Single 10 = Block 11 = Cascade		Address count: 0 = Increment 1 = Decrement	Auto-initialize: 0 = Disable 1 = Enable	Transfer select: 00 = Verify 10 = Mem. read 01 = Mem. write 11 = Reserved		Channel select: 00 = Ch. 4 10 = Ch. 6 01 = Ch. 5 11 = Ch. 7	
Port 0D2h	**DMAC2 DMA Request Register**						
Reserved: Write as 0.					Request: 0 = Clear 1 = Set	Channel Select: 00 = Ch. 4 10 = Ch. 6 01 = Ch. 5 11 = Ch. 7	
Port 0D0h	**DMAC2 Command Register**						
DACK active sense: 0 = Low 1 = High	DRQ active sense: 0 = High 1 = Low	Extended write: 0 = Disable 1 = Enable	Rotating priority: 0 = Disable 1 = Enable	Compressed timing: 0 = Disable 1 = Enable	DMAC operation: 0 = Enable 1 = Disable	Channel 0 address hold: 0 = Disable 1 = Enable	Memory-to-memory: 0 = Disable 1 = Enable
Port 0DEh	**DMAC2 Mask Register**						
Reserved: Write as 0.				Channel 7: 0 = Unmasked 1 = Masked	Channel 6: 0 = Unmasked 1 = Masked	Channel 5: 0 = Unmasked 1 = Masked	Channel 4: 0 = Unmasked 1 = Masked

Table 4-65 DMA Commands

Command	Function	Command Port Address: DMA Channels 7-5	Command Port Address: DMA Channels 3-0
Set Single Mask Bits Register	Sets or clears individual mask register bits without having to do a read/modify/write of the Mask Register.	Write 0D4h: Bits [1:0] select the channel, bit 2 selects the new mask bit value.	Write 00Ah: Bits [1:0] select the channel, bit 2 selects the new mask bit value.
Clear Mask	Unmasks all DMA channels at once.	Write any value to 0DCh.	Write any value to 00Eh.
Reset Mode Register Readback Counter	Resets the Mode Register readback function to start at register 0. The next four Mode Register reads then return Channels 0, 1, 2, and 3 for that DMAC.	Read 0DCh (then read 0D6h four times to get the Mode Register values).	Read 00Eh (then read 00Bh four times to get the Mode Register values).
Master Clear	Clears all values, masks all channels, just like a hardware reset.	Write any value to 0DAh.	Write any value to 00Dh.
Clear Byte Pointer Flip-Flop	Resets the byte pointer flip-flop so that the next byte access to a word-wide DMA register is to the low byte.	Write any value to 0D8h.	Write any value to 00Ch.
Set Byte Pointer Flip-Flop	Sets the byte pointer flip-flop so that the next byte access to a word-wide DMA register is to the high byte.	Read 0D8h.	Read 00Ch.

4.10.4 Timer Programming Registers

The integrated IPC provides an 8254-type timer with three channels that is register compatible with AT-type systems. The registers of this logic module are listed below. These registers are accessed directly through the I/O subsystem (no index/data method is used). Table 4-66 lists the register locations.

Table 4-66 Timer Control and Status Registers

Name	Function	Port Address Timer Channel 0	Channel 1	Channel 2
Counter Registers Access	Used to write and read the word-wide count. Writes always program the base value. Reads return either the instantaneous count value or the latched count value.	040h	041h	042h
Counter Mode Command	Selects the operational mode for each timer counter.	Write 043h		
Counter Latch Command	Latches the count from the selected register for reading at the associated counter register access port.	Write 043h then read 040h, 041h, and/or 042h		
Readback Command	Selects whether count or status, or both, will be latched for subsequent reading at the associated counter register access port. If both are selected, status is returned first. This command can latch information from more than one counter at a time.	Write 043h then read 040h, 041h, and/or 042h		

Table 4-67 Timer Control Bits

7	6	5	4	3	2	1	0
Port 043h			**Counter Mode Command (WO)**				
Counter select: 00 = Counter 0 01 = Counter 1 10 = Counter 2 11 = Readback command (see below)		Counter access: 00 = Counter latch command (see below) 01 = R/W LSB only 10 = R/W MSB only 11 = R/W LSB followed by MSB		Mode select: 000 = M0) Interrupt on terminal count 001 = M1) Hardware retrig. one-shot X10 = M2) Rate generator X11 = M3) Square wave generator 100 = M4) Software-triggered strobe 101 = M5) Hardware-triggered strobe			Count mode select: 0 = 16-bit binary 1 = 4-decade BCD
Port 043h			**Counter Latch Command (WO)**				
Counter select: 00 = Counter 0 01 = Counter 1 10 = Counter 2 11 = Illegal		Counter latch command = 00		Don't care			
Port 043h			**Readback Command (WO)**				
Readback command = 11		Latch count: 0 = Yes 1 = No	Latch status: 0 = Yes 1 = No	Counter 2 select: 0 = Yes 1 = No	Counter 1 select: 0 = Yes 1 = No	Counter 0 select: 0 = Yes 1 = No	Reserved: Write as 0.
Port 043h			**Status Byte (RO)**				
OUT signal status	Null count, counter contents valid: 0 = Yes 1 = No (being updated)	Return bits [5:0] written in Counter Mode Command (see above)					

4.10.4.1 Shadow Registers To Support Timer

Values written to the timer are not always directly readable in the AT architecture. However, FireStar shadows these values as they are written so that they can be read back later through the configuration registers. The values from index 90h to 96h are valid only when a Counter Mode Command byte for the counter has been written to the timer register at I/O Port 043h. Setting bits 043h[5:4] = 11 starts the sequence.

4.10.5 Writing/Reading I/O Port 070h

The AT architecture does not allow the readback of the NMI enable bit settings and the RTC index value written at I/O Port 070h. However, the FireStar logic makes the NMI Enable bit setting, along with the last RTC index value written to I/O Port 070h, available for reading in its shadow register set.

4.10.5.1 RTC Index Shadow Register

This shadow register is read as a normal FireStar configuration register: write 98h to I/O Port 022h followed immediately by an I/O read at I/O Port 024h.

Table 4-68 Timer Support Shadow Registers

7	6	5	4	3	2	1	0
Index 90h			**Channel 0 Low Byte**				
- Timer Channel 0 count low byte, A[7:0]							
Index 91h			**Channel 0 High Byte**				
- Timer Channel 0 count high byte, A[15:8]							
Index 92h			**Channel 1 Low Byte**				
- Timer Channel 1 count low byte, A[7:0]							
Index 93h			**Channel 1 High Byte**				
- Timer Channel 1 count high byte, A[15:8]							
Index 94h			**Channel 2 Low Byte**				
- Timer Channel 2 count low byte, A[7:0]							
Index 95h			**Channel 2 High Byte**				
- Timer Channel 2 count high byte, A[15:8]							
Index 96h			**Write Counter High/Low Byte Latch**				
Unused		Ch. 2 read LSB toggle bit	Ch. 1 read LSB toggle bit	Ch. 0 read LSB toggle bit	Ch. 2 write LSB toggle bit	Ch. 1 write LSB toggle bit	Ch. 0 write LSB toggle bit

Table 4-69 RTC Index Register - I/O Port 070h (WO)

7	6	5	4	3	2	1	0
NMI Enable: 0 = Disable 1 = Enable	RTC/CMOS RAM Index						

Table 4-70 RTC Index Shadow Register - Index 98h (RO)

7	6	5	4	3	2	1	0
NMI Enable Setting	CMOS RAM Index Last Written						

4.10.6 IRQ8 Polarity

The recognition of the IRQ8 interrupt can be inverted through SYSCFG 50h[5]. In the normal AT architecture, IRQ8 is active low and driven by an open-collector output of the RTC against a pull-up resistor.

4.10.7 Fast GATEA20 and Reset Emulation

FireStar will intercept commands to Ports 060h and 064h so that it can emulate the keyboard controller, allowing the generation of the fast GATEA20 and fast INIT signals. The decode sequence is software transparent and requires no BIOS modifications to function. The fast GATEA20 generation sequence involves writing "D1h" to Port 064h, then writing data "02h" to Port 060h. The fast CPU "warm reset" function is generated when a Port 064h write cycle with data "FEh" is decoded. A write to Port 064h with data "D0h" will enable the status of GATEA20 (bit 1 of Port 060h) and the warm reset (bit 0 of Port 060h) to be readable.

If keyboard emulation has been disabled (PCIDV1 41h[4] = 1), FireStar will still intercept reset and GATEA20 commands to Port 092h and generate CPUINIT to the CPU. However, FireStar has to be programmed to accept the KBRST# and KBA20M signals from the keyboard controller to generate CPUINIT and A20M# to the CPU because it will not intercept Port 060/064h commands. Figure 4-26 shows the connectivity when keyboard emulation has been disabled.

Table 4-71 IRQ8 Polarity Bit - SYSCFG 50h

7	6	5	4	3	2	1	0
SYSCFG 50h			**PMU Control Register 4**				**Default = 00h**
		IRQ8 polarity: 0 = Active low 1 = Active high					

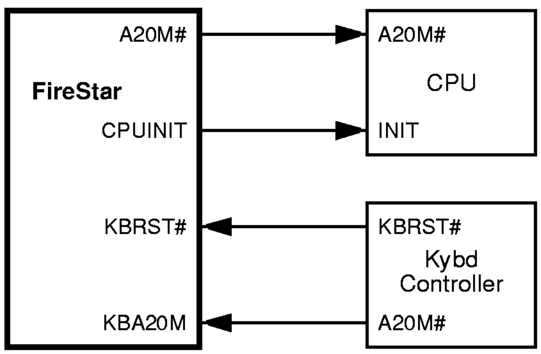
Table 4-72 Fast GATEA20 and Reset Emulation Related I/O Port Registers

7	6	5	4	3	2	1	0
Port 061h			**Port B Register**				
System parity check (RO)	I/O channel check (RO)	Timer OUT2 detect (RO)	Refresh detect (RO)	I/O channel check: 0 = Enable 1 = Disable	Parity check: 0 = Enable 1 = Disable	Speaker output: 0 = Disable 1 = Enable	Timer 2 Gate: 0 = Disable (from CPU address) 1 = Enable
Port 060h & 064h			**Keyboard I/O Control Registers**				
The 82C700 will intercept commands to Ports 060h and 064h so that it may emulate the keyboard controller, allowing the generation of the fast GATEA20 and fast CPURST signals. The decode sequence is software transparent and requires no BIOS modifications to function. The fast GATEA20 generation sequence involves writing 'D1h' to Port 64h, then writing data '02h' to Port 060h. The fast CPU warm reset function is generated when a Port 064h write cycle with data 'FEh ' is decoded. A write to Port 064h with data D0h will enable the status of GATEA20 (bit 1 of Port 060h) and the system reset (bit 0 of Port 060h) to be readable							
Port 092h			**PS/2 Reset Control Registers**				**Default = 00h**
Reserved						A20M#: 0 = A20M# active 1 = A20M# inactive	Fast Reset (automatically clears back to 0): 1 = INIT sent to the 3.3V CPU

Table 4-73 Keyboard Emulation Disable Bit

7	6	5	4	3	2	1	0
PCIDV1 41h			**Keyboard Controller Select Register**				**Default = 00h**
RDKBDPRT (RO): Keyboard controller has received Command D0h and has not received the following 060h read.	WRKBDPRT (RO): Keyboard controller has received Command D1h and has not received the following 060h write.	IMMINIT: Generate INIT immediately on FEh Command. 0 = Generate INIT immediately on FEh Command 1 = Wait for halt before INIT for keyboard reset	KBDEMU: Keyboard emulation 0 = Enable 1 = Disable	KBDCS# includes Port 062h and 066h: 0 = Disable 1 = Enable	Reserved		

Figure 4-26 Connections with Keyboard Emulation Disabled



4.11 Integrated Local Bus Enhanced IDE Interface

The IDE controller in FireStar is based on OPTi's Bus Master PCI IDE Module (MIDE) which is designed as a fast and flexible interface between the PCI bus and two channels of IDE devices (up to four devices). Each channel supports an integrated 8-level (32-byte) read prefetch FIFO and an 8-level (32-byte) posted write FIFO for bus mastering burst read and write operations on the PCI bus, substantially improving the performance over the typical slave IDE implementations. The Enhanced ATA Specification can be supported by programming the internal registers up to IDE PIO Mode 5 and Single- and/or Multi-Word DMA Mode 2 timing. The module is designed to be backward compatible to the Viper-N+ IDE interface.

When the internal IDE controller is disabled, FireStar passes the IDE cycles to the ISA bus if the cycles are not claimed on the PCI bus.

4.11.1 Performance and Power

Enhanced IDE uses the SD bus for its data transfers, but does not have to use slow ISA bus transfers because of its dedicated HDRD#, HDWR#, and DBE# signals. Essentially, the local bus IDE controller can run extremely short cycles because all timing aspects of the cycle are directly programmable to meet the capabilities of the drive being used.

FireStar's implementation of local bus IDE is designed to save power. The buffers to/from the IDE are tristated between cycles. Therefore, no power is wasted toggling the IDE data lines when the IDE is not in use.

4.11.2 Bus Mastering IDE Controller

FireStar features a new bus mastering IDE controller interface. Multiplexed operation allows the chip to very efficiently use only two dedicated pins (or four for four-drive bus mastering) yet still allows IDE operation that is fully concurrent with every other high-speed system activity.

The IDE controller operates in either programmed I/O (PIO) mode, bus mastering mode, or emulated bus mastering mode. In both cases, the control signals are multiplexed onto the XD[7:0] lines and the data is presented on the SD[15:0] bus. External buffers may be required to interface these signals to the IDE drive, but can be eliminated in certain designs.

Dedicated signal DBEW#, and optional signals DBEX#, DBEY#, and DBEZ# (on PIO pins), are provided to enable separate sets of buffers for each of the two supported drive channels (two drives per channel). The drive read and write commands come from the XD bus pins, qualified by DBE#. PCIDV1 AEh and AFh select the decoding that will take place at each DBE# pin (see Table 4-74).

Decoding for cable 0 can be disabled completely through PCIDV1 4Fh[5] if only cable 1 is used locally (for example, if a docking station is connected and system boot should occur from the docking station drive instead of from the local drive).

Bus mastering requires the addition of the DDRQ and DDACK# signal pair for each drive cable. The DDRQ0-1 signals are supported as separate inputs on the chip; DDACK0-1# are multiplexed onto the XD lines like the other control lines, since they are meaningful only when a cycle is taking place (DBE# signal active).

Table 4-74 IDE Pin Programming Registers

7	6	5	4	3	2	1	0
PCIDV1 AEh			**DBE# Select Register 1**				**Default = 01h**
Reserved	DBEX# selection: 000 = Disable (Default) 001 = DBE0#: Cable 0, Drives 0 and 1 010 = DBE0-0#: Cable 0, Drive 0 011 = DBE0-1#: Cable 0, Drive 1 100 = Decode all IDE accesses 101 = DBE1#: Cable 1, Drives 0 and 1 110 = DBE1-0#: Cable 1, Drive 0 111 = DBE1-1#: Cable 1, Drive 1			Reserved	DBEW# selection: 000 = Disable 001 = DBE0#: Cable 0, Drives 0 and 1(Default) 010 = DBE0-0#: Cable 0, Drive 0 011 = DBE0-1#: Cable 0, Drive 1 100 = Decode all IDE accesses 101 = DBE1#: Cable 1, Drives 0 and 1 110 = DBE1-0#: Cable 1, Drive 0 111 = DBE1-1#: Cable 1, Drive 1		

Table 4-74 IDE Pin Programming Registers

7	6	5	4	3	2	1	0
PCIDV1 AFh			**DBE# Select Register 2**				**Default = 00h**
Reserved	DBEZ# selection: 000 = Disable (Default) 001 = DBE0#: Cable 0, Drives 0 and 1 010 = DBE0-0#: Cable 0, Drive 0 011 = DBE0-1#: Cable 0, Drive 1 100 = Decode all IDE accesses 101 = DBE1#: Cable 1, Drives 0 and 1 110 = DBE1-0#: Cable 1, Drive 0 111 = DBE1-1#: Cable 1, Drive 1			Reserved	DBEY# selection: 000 = Disable (Default) 001 = DBE0#: Cable 0, Drives 0 and 1 010 = DBE0-0#: Cable 0, Drive 0 011 = DBE0-1#: Cable 0, Drive 1 100 = Decode all IDE accesses 101 = DBE1#: Cable 1, Drives 0 and 1 110 = DBE1-0#: Cable 1, Drive 0 111 = DBE1-1#: Cable 1, Drive 1		
PCIDV1 4Fh			**Miscellaneous Control Register C - Byte 1**				**Default = 20h**
		Primary IDE interface (1F0h): 0 = Disable 1 = Enable (Default)					

4.11.2.1 Isolation of Drives

Most notebook designs require that each IDE drive on a cable be capable of individual power-down without affecting other drives in the system. For example, an IDE CD-ROM and IDE hard drive that share the same cable could be power-managed separately to avoid having to keep the CD-ROM alive while the hard drive is active (or vice-versa).

The FireStar solution includes programmable pin options described below to allow easier isolation in typical implementations.

4.11.2.2 IDE Control Pinout Options

FireStar provides several programmable pin options to optimize the system design and reduce the need for external TTL. Refer to Section 3.3, "Programmable I/O Pins" for information on assigning these pin functions.

- **DBE0A/B# and DBE1A/B#**
  FireStar can be programmed to generate separate buffer enable signals for each drive on the cable. Normally each cable has its own buffer enable: DBE0# for cable 0, decoded from I/O ports 1F0-7+3F6-7h; and DBE1# for cable 1, decoded from I/O ports 170-7+376-7h. The A/B# feature takes this decoding one step further and selects "drive A" or "drive B" on the cable according to the value last written to bit 1F6h[4] for cable 0, or 176h[4] for cable 1.
- **Dedicated DDACK#**
  Bus mastering IDE drives must use one DDRQ/DDACK# pair per cable. The standard FireStar pinout provides DDRQ as dedicated inputs, but uses an XD bus pin qualified by DBE# to drive DDACK# to the drive. The total number of signals controlled by DBE# in this case is 9, a very inconvenient value for use with 8-bit TTL. Therefore, FireStar allows for up to two dedicated DDACK# pins, one for each cable.
- **Dedicated DRD#, DWR#, DCS1#, DCS3#, DA[2:0]**
  In a small system, unused pins can be replaced with IDE control signals. This feature allows the designer to avoid using any TTL to support the IDE. This solution is especially well suited for an ISA-less system, so that the SD[15:0] bus can be used solely to support the IDE drives.

Figure 4-27 illustrates the connections typically required for a fully-isolated IDE drive.

Figure 4-27 IDE Interface Using Individual TTL

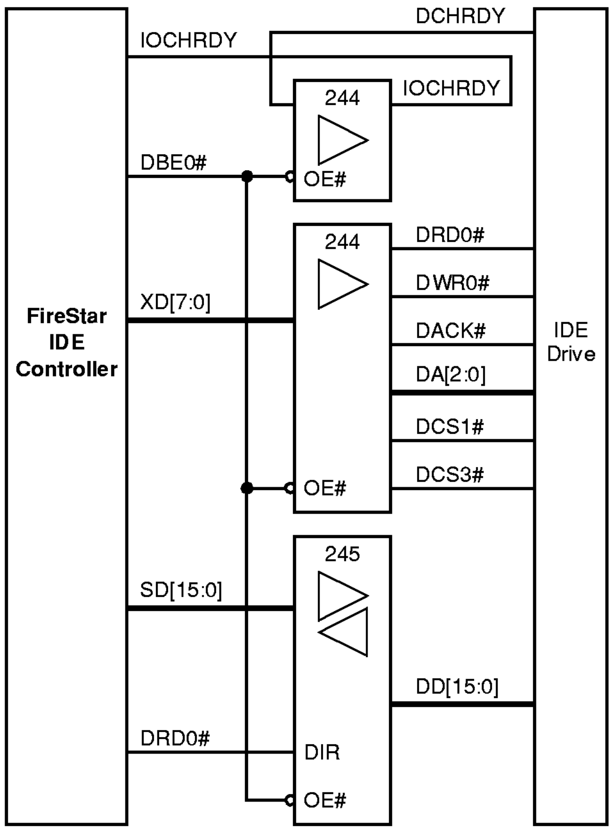


Figure 4-28 IDE Interface Using Zero-TTL

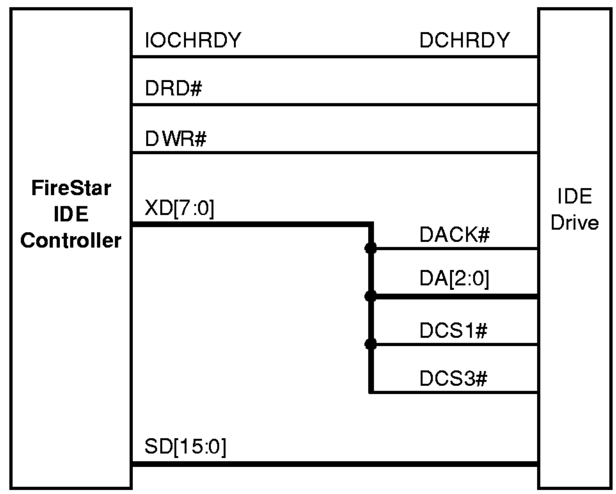


Figure 4-28 shows an implementation for a minimal system without separately buffered drives. A typical notebook design with few ISA-bus devices could use this approach. Note that PIO pins have been assigned IDE control functions DRD# and DWR# to allow a zero-TTL solution, yet some of the control signals still come from the XD[7:0] bus. The X-bus control lines will also toggle during cycles to the ROM, RTC, and KBC but will not have an effect on the IDE drive since DRD# and DWR# are inactive. If this situation is not acceptable and there are enough PIO pins free, all IDE control signals can be assigned to dedicated pins. Table 4-75 list the general IDE control line assignment.

Table 4-75 General IDE Control Line Assignment

Primary Pin Name	IDE Signal	Description
SD[15:0]	DD[15:0]	IDE data bus
XD7	DCS3#	IDE chip select for 3F6-7h or 376-7h I/O access
XD6	DCS1#	IDE chip select for 1F0-7h or 170-7h I/O access
XD5	DDACK#	Bus master IDE DMA acknowledge
XD4	DA2	IDE address
XD3	DA1	
XD2	DA0	
XD1	DRD#	IDE drive read command line, qualified by DBE#
XD0	DWR#	IDE drive write command line, qualified by DBE#
Dedicated pin	DBE0#	Command buffer enable to IDE cable 0
Dedicated pin	DDRQ0	Bus mastering IDE drive DMA request line for cable 0
PIO pin	DBE1#	Command buffer enable to IDE cable 1
PIO pin	DDRQ1	Bus mastering IDE drive DMA request line for cable 1

4.11.2.3 Dedicated Secondary IDE Interface

In a true ISA-less system, where even the ROM is handled by an external device on the PCI bus, a large number of ISA support pins become available. These pins can be reassigned to independently support a second IDE drive with no external TTL. The primary ISA interface is also available to directly support a drive in this case. Either physical interface can be programmed to respond as:

- Cable 0 Drive 0
- Cable 0 Drive 1
- Cable 1 Drive 0
- Cable 1 Drive 1

In No-ISA Mode, all SD, XD, SA, and ISA command signals stay low at reset so as not to drive signals to a possibly non powered drive. Table 4-76 lists the ISA pins that are reassigned to the primary and secondary IDE drive function.

Table 4-76 Control Line Assignment for two IDE Cables (No-ISA Mode)

Original Pin Name Drives Cable 0	Original Pin Name Drives Cable 1	IDE Signal Name	Description
SD[15:0]	SA[15:0]	DD[15:0]	IDE data bus
XD7	MRD#	DCS3#	IDE chip select for 3F6-7h or 376-7h I/O access
XD6	MWR#	DCS1#	IDE chip select for 1F0-7h or 170-7h I/O access
XD5	DBE0#	DDACK#	Bus master IDE DMA acknowledge
XD4	RTCWR#	DA2	IDE address
XD3	RTCRD#	DA1	
XD2	RTCAS	DA0	
XD1	IORD#	DRD#	IDE drive read command line
XD0	IOWR#	DWR#	IDE drive write command line
DDRQ0	PIOx	DDRQ	Bus master IDE drive DMA request line
IOCHRDY	PIOx	DCHRDY	Drive channel ready line

4.11.3 Programming the IDE Controller

The IDE controller has four register spaces that control it:

1. SYSCFG - SYSCFG registers are accessed by writing Port 022h with an index and writing or reading from Port 024h with a data value.
2. PCIIDE - The PCIIDE space is accessed through PCI Configuration Mechanism #1 by addressing Bus #0, Device #14h, Function #0.
3. IDE I/O - The IDE I/O space is a register set that is hidden behind the IDE I/O ports, and is normally not accessible. A special sequence must be followed for enabling/disabling access to these registers. Unless otherwise noted, all references to IDE I/O space in this section pertain to this hidden space. These configuration registers share the I/O ports.
4. Bus Master IDE registers - Mapped to system I/O space.

References to the IDE I/O space 10Fh-1F7h are used below. The same concepts apply to the 170h-177h space.

4.11.3.1 Enabling Access to IDE I/O Space

To enable access to the IDE I/O register space, the following procedure must be followed:

- Perform two consecutive 16-bit reads from I/O Port 1F1h. This operation makes the register space accessible for the next I/O operation.
- Perform an 8-bit write to I/O Port 1F2h with a value of 03h. This programming keeps the registers accessible indefinitely.

4.11.3.2 Disabling Access to IDE I/O Space

After enabling and programming the IDE I/O registers, these registers must be hidden from standard access before IDE operations can begin. Two options are available:

- Write Port 1F2h with a value of C3h to disable the IDE I/O register space and fully enable IDE operation, and also prevent any future access to this space until the next hardware reset.

- Write Port 1F2h with a value of 83h to disable the IDE I/O space and fully enable IDE operation, but leave open the future possibility of accessing this space by enabling the space again as previously described in Section 4.11.3.1, "Enabling Access to IDE I/O Space".

Table 4-77 shows the registers associated with enabling and disabling access to the IDE I/O space.

Table 4-77 Enabling/Disabling Access to IDE I/O Space Registers

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">I/O Address 1F1h</td><td colspan="4">Write Cycle Timing Register - Timing 0[1]</td><td colspan="2">Default = xxh</td></tr>
<tr><td colspan="4">Write pulse width:
The value programmed in this register plus one determines the DWR# pulse width in PCICLKs (for a 16-bit write from the IDE Data Register). See Table 4-82 or Table 4-83.
(Default = xxxx)</td><td colspan="4">Write recovery time:
The value programmed in this register plus two determines the recovery time between the end of DWR# and the next DA[2:0]/DCSx# being presented (after a 16-bit write from the IDE Data Register), measured in PCICLKs. See Table 4-82 or Table 4-83.
(Default = xxxx)</td></tr>
<tr><td colspan="8">(1) Timing 0 can be programmed only if IDE I/O 1F6h[0] = 0. The timing programmed into this register is applied for IDE accesses to drives as selected by 1F3h[3:2] and 1F3h[7].</td></tr>
<tr><td colspan="2">I/O Address 1F1h</td><td colspan="4">Write Cycle Timing Register - Timing 1[1]</td><td colspan="2">Default = xxh</td></tr>
<tr><td colspan="4">Write pulse width:
The value programmed in this register plus one determines the DWR# pulse width in PCICLKs (for a 16-bit write from the IDE Data Register). See Table 4-82 or Table 4-83.
(Default = xxxx)</td><td colspan="4">Write recovery time:
The value programmed in this register plus two determines the recovery time between the end of DWR# and the next DA[2:0]/DCSx# being presented (after a 16-bit write from the IDE Data Register), measured in PCICLKs. See Table 4-82 or Table 4-83.
(Default = xxxx)</td></tr>
<tr><td colspan="8">(1) Timing 1 can be programmed only if IDE I/O 1F6h[0] = 1. The timing programmed into this register is applied for IDE accesses to drives as selected by 1F3h[3:2] and 1F3h[7].</td></tr>
<tr><td colspan="2">I/O Address 1F2h</td><td colspan="4">Internal ID Register</td><td colspan="2">Default = xxh</td></tr>
<tr><td>Configuration disable (WO):
0 = Enable accesses to internal IDE controller registers
1 = Disable accesses to internal IDE controller registers until another 2 consecutive I/O reads from 1F1h.
(Default = 1)</td><td>Configuration off (WO):
0 = Enable accesses to internal IDE controller registers
1 = Disable all accesses to internal IDE controller registers until power-down or reset.</td><td colspan="4">Reserved (RO): Write to 0.
(Default = xxxx)</td><td colspan="2">Reserved:
Must be written 11. If not, all writes to the IDE I/O Registers will be blocked.</td></tr>
</table>

4.11.3.3 Setting up the IDE Controller

The steps described below must be followed prior to initializing the timing for the drives, for both PIO and bus mastering capability.

1. Configure the PCI IDE module as PCI Device #14h, Function #0, by setting SYSCFG ADh[4] = 0.
2. Set the PCI bus frequency in IDE I/O 1F5h[0].
3. The 32-byte read prefetch FIFO should be enabled at PCIIDE 40h[5].
4. Concurrent refresh and IDE cycles should be enabled at PCIDV1 52h[0].
5. PCI IDE one wait state reads for primary and secondary channels should be enabled at IDE I/O Register 1F3h[4].
6. Read prefetch for primary and secondary channels should be enabled at IDE I/O Register 1F6h[6].
7. Enable master capability at PCIIDE 04h[2].
8. Assign a non-conflicting I/O address for bus master IDE base address in PCIIDE 20h-23h. The I/O address must be less than 64KB.

Table 4-78 shows the register bits used for setting up the IDE controller.

Table 4-78 IDE Interface Control Registers

7	6	5	4	3	2	1	0
SYSCFG ADh			**Feature Control Register 3**				**Default = 00h**
			PCIIDE responds as: 0 = Device 14h, Function 0 1 = Device 01h, Function 1				
I/O Address 1F5h			**Strap Register**				**Default = xxh**
							PCI CLK speed: 0 = 33MHz 1 = 25MHz
PCIIDE 40h			**IDE Initialization Control Register**				**Default = 00h**
		Enhanced Slave: 0 = 82C621A-compatible mode, uses a 16-byte FIFO in PIO Mode 1 = Enhanced mode, uses a 32-byte FIFO in PIO Mode		Secondary IDE: 0 = Enable 1 = Disable This bit is effective only if PCIDV1 4Fh[6] = 1.			

Table 4-78 IDE Interface Control Registers (cont.)

7	6	5	4	3	2	1	0
PCIDV1 52h			**Misc. Controller Register 3**				**Default = 00h**
							Concurrent refresh and IDE cycle: 0 = Disable 1 = Enable ISA devices that rely on accurate refresh addresses for proper operation should disable this bit.
I/O Address 1F3h			**Control Register**				**Default = xxh**
			Enable one wait state read: 0 = 2 WS minimum 1 = 1 WS minimum for data reads				
I/O Address 1F6h			**Miscellaneous Register**				**Default = xxh**
	Read prefetch: 0 = Disable 1 = Enable						
PCIIDE 04h			**Command Register - Byte 0**				**Default = 45h**
					IDE controller becomes a PCI master to generate PCI accesses: 0 = Disable 1 = Enable		
PCIIDE 20h-23h			**Bus Master IDE Base Address Register**				**Default = 00000001h**
This register is the I/O base address indicator for the Bus Master IDE Registers. The address block has a size of 16 bytes. - Bits [3:0] are read-only and default to 0001. - Bits [31:4] are writable.							

4.11.4 Programming Timing Information

The FireStar IDE controller of supports up to four IDE drives on two cables with independent timing requirements. After common or independent timing for all the drives is programmed, the IDE controller core tracks application software accesses to the IDE ports and automatically enables the programmed independent timing for the drive being accessed. Figure 4-29 shows the configuration for the IDE controller while operating in PIO Mode. Figure 4-30 depicts the timing parameters associated with PIO Mode drives.

A variety of options exist for programming the IDE timing for different drives. It is possible to program common PIO mode timing for all detected drives, customize independent timing for each drive, or enable bus mastering support on a drive-by-drive basis. Follow either Section 4.11.4.1, "Enabling Common Timing for All Drives", or Section 4.11.4.2, "Enabling Independent Timing", to setup global timing or independent timing information for the drives.

Figure 4-29 PIO Mode Configuration

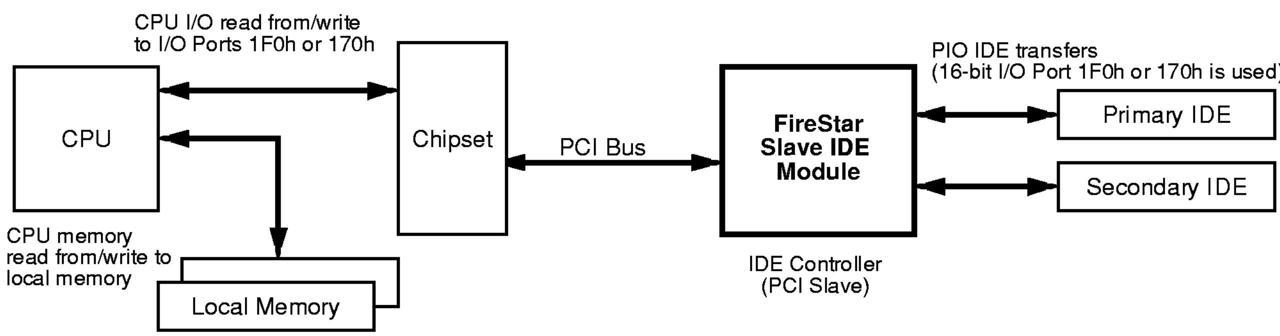
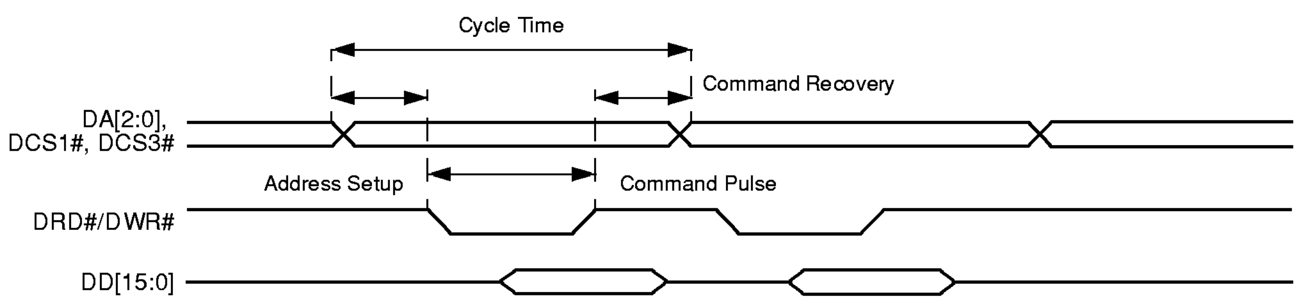


Figure 4-30 PIO Mode Cycle Timing



4.11.4.1 Enabling Common Timing for All Drives

Table 4-79 shows the registers associated with programming common timing for all enabled drives. Common IDE timing for all drives is enabled by the IDE controller until independent timing is programmed.

The default common timing for all drives is PIO Mode 0. Common PIO Modes 0-3 timing for all enabled drives can be programmed by setting PCIIDE 40h[1:0].

To enable common PIO Mode 4 or Mode 5 timing, first ensure that PIO Mode 3 is set up in PCIIDE 40h[1:0], as described above. Subsequently, the appropriate bits in PCIIDE 43h[7:0] can be set to enable Mode 4 or Mode 5 timing.

Table 4-79 IDE Timing Control "Common Timing"

7	6	5	4	3	2	1	0
PCIIDE 40h			**IDE Initialization Control Register**				**Default = 00h**
						Mode: These bits control the default 16-bit cycle times for all IDE devices and can be overridden by programming the IDE I/O Registers. 00 = $\geq$ 600ns cycle time (PIO Mode 0) 01 = $\geq$ 383ns cycle time (PIO Mode 2) 10 = $\geq$ 240ns cycle time (PIO Mode 1) 11 = $\geq$ 180ns cycle time (PIO Mode 3) These bits are effective only if PCIDV1 4Fh[6] = 1.	
PCIIDE 43h			**IDE Enhanced Mode Register**				**Default = 00h**
Enhanced Mode for Drive 1 on Secondary Channel: Sets 16-bit cycle times for IDE PIO Modes 4 and 5 or Multi-Word DMA Modes 1 and 2. 00 = Disabled, control by corresponding Timing Registers Set 01 = PIO Mode 4 or Multi-Word DMA Mode 1, command inactive for 2 PCICLKs 10 = PIO Mode 5 or Multi-Word DMA Mode 2, command inactive for 1 PCICLK 11 = Reserved Corresponding 170h/171h[3:0] must be set to 0 before these two bits are set to 01 or 10.		Enhanced Mode for Drive 0 on Secondary Channel: Sets 16-bit cycle times for IDE PIO Modes 4 and 5 or Multi-Word DMA Modes 1 and 2. 00 = Disabled, control by corresponding Timing Registers Set 01 = PIO Mode 4 or Multi-Word DMA Mode 1, command inactive for 2 PCICLKs 10 = PIO Mode 5 or Multi-Word DMA Mode 2, command inactive for 1 PCICLK 11 = Reserved Corresponding 170h/171h[3:0] must be set to 0 before these two bits are set to 01 or 10.		Enhanced Mode for Drive 1 on Primary Channel: Sets 16-bit cycle times for IDE PIO Modes 4 and 5 or Multi-Word DMA Modes 1 and 2. 00 = Disabled, control by corresponding Timing Registers Set 01 = PIO Mode 4 or Multi-Word DMA Mode 1, command inactive for 2 PCICLKs 10 = PIO Mode 5 or Multi-Word DMA Mode 2, command inactive for 1 PCICLK 11 = Reserved Corresponding 1F0h/1F1h[3:0] must be set to 0 before these two bits are set to 01 or 10.		Enhanced Mode for Drive 0 on Primary Channel Sets 16-bit cycle times for IDE PIO Modes 4 and 5 or Multi-Word DMA mode 1 and 2. 00 = Disabled, control by corresponding Timing Registers Set 01 = PIO Mode 4 or Multi-Word DMA Mode 1, command inactive for 2 PCICLKs 10 = PIO Mode 5 or Multi-Word DMA Mode 2, command inactive for 1 PCICLK 11 = Reserved Corresponding 1F0h/1F1h[3:0] must be set to 0 before these two bits are set to 01 or 10.	

4.11.4.2 Enabling Independent Timing

If required, independent timing can be programmed for each drive in each channel, by accessing the IDE I/O register space. For every enabled drive, three timing choices are available: the common timing described earlier, Timing 0 or Timing 1. Timing 0 and Timing 1 are two sets of timing that provide separate read/write pulse widths, read/write recovery times, common address setup time, and common channel ready hold times. The relevant timing registers for the primary channel are listed in Table 4-80.

Table 4-80 IDE Timing Control "Independent Timing"

7	6	5	4	3	2	1	0
I/O Address 1F0h			**Read Cycle Timing Register - Timing 0(1)**				**Default = xxh**
Read pulse width: The value programmed in this register plus one determines the DRD# pulse width in PCICLKs (for a 16-bit read from the IDE Data Register). See Table 4-82 or Table 4-83. (Default = xxxx)				Read recovery time: The value programmed in this register plus two determines the recovery time between the end of DRD# and the next DA[2:0]/DCSx# being presented (after a 16-bit read from the IDE Data Register), measured in PCICLKs. See Table 4-82 or Table 4-83. (Default = xxxx)			
(1) Timing 0 can be programmed only if IDE I/O 1F6h[0] = 0. The timing programmed into this register is applied for IDE accesses to drives as selected by 1F3h[3:2] and 1F3h[7].							
I/O Address 1F0h			**Read Cycle Timing Register - Timing 1(1)**				**Default = xxh**
Read pulse width: The value programmed in this register plus one determines the DRD# pulse width in PCICLKs (for a 16-bit read from the IDE Data Register). See Table 4-82 or Table 4-83. (Default = xxxx)				Read recovery time: The value programmed in this register plus two determines the recovery time between the end of DRD# and the next DA[2:0]/DCSx# being presented (after a 16-bit read from the IDE Data Register), measured in PCICLKs. See Table 4-82 or Table 4-83. (Default = xxxx)			
(1) Timing 1 can be programmed only if IDE I/O 1F6h[0] = 1. The timing programmed into this register is applied for IDE accesses to drives as selected by 1F3h[3:2] and 1F3h[7].							
I/O Address 1F1h			**Write Cycle Timing Register - Timing 0(1)**				**Default = xxh**
Write pulse width: The value programmed in this register plus one determines the DWR# pulse width in PCICLKs (for a 16-bit write from the IDE Data Register). See Table 4-82 or Table 4-83. (Default = xxxx)				Write recovery time: The value programmed in this register plus two determines the recovery time between the end of DWR# and the next DA[2:0]/DCSx# being presented (after a 16-bit write from the IDE Data Register), measured in PCICLKs. See Table 4-82 or Table 4-83. (Default = xxxx)			
(1) Timing 0 can be programmed only if IDE I/O 1F6h[0] = 0. The timing programmed into this register is applied for IDE accesses to drives as selected by 1F3h[3:2] and 1F3h[7].							
I/O Address 1F1h			**Write Cycle Timing Register - Timing 1(1)**				**Default = xxh**
Write pulse width: The value programmed in this register plus one determines the DWR# pulse width in PCICLKs (for a 16-bit write from the IDE Data Register). See Table 4-82 or Table 4-83. (Default = xxxx)				Write recovery time: The value programmed in this register plus two determines the recovery time between the end of DWR# and the next DA[2:0]/DCSx# being presented (after a 16-bit write from the IDE Data Register), measured in PCICLKs. See Table 4-82 or Table 4-83. (Default = xxxx)			
(1) Timing 1 can be programmed only if IDE I/O 1F6h[0] = 1. The timing programmed into this register is applied for IDE accesses to drives as selected by 1F3h[3:2] and 1F3h[7].							
Note: Both Timing 0 and Timing 1 sets share the same address setup and DRDY delay times as programmed in 1F6h[5:4] and 1F6h[3:2].							

Table 4-80 IDE Timing Control "Independent Timing" (cont.)

7	6	5	4	3	2	1	0
I/O Address 1F3h			**Control Register**				**Default = xxh**
Timing register value select: 0 = Basic 1 = Enhanced				Drive 1 timing select: Basic (1F3h[7] = 0): 0 = Determined by PCIIDE 40h[1:0] 1 = Timing 1 Enhanced (1F3h[7] = 1): 0 = Timing 1 1 = Timing 0	Drive 0 timing select: Basic (1F3h[7] = 0): 0 = Determined by PCIIDE 40h[1:0] 1 = Timing 1 Enhanced (1F3h[7] = 1): 0 = Timing 1 1 = Timing 0		
Note: Bits 2, 3 and 7 of the Control Register should be enabled after the Cycle Timing Registers and Miscellaneous Register are programmed. See Table 4-81 for programming options.							
I/O Address 1F6h			**Miscellaneous Register**				**Default = xxh**
		Address setup time:(1) The value programmed in this register plus one determines the address setup time between DRD# or DWR# going active and DA[2:0], DCS3#, DCS1# being presented, measured in PCICLKs. See Table 4-82 or Table 4-83. (Default = xx)		Delay:(1) The value programmed in this register plus two determines the minimum number of PCICLKs between DRDY# going high and DRD# or DWR# going inactive. See Table 4-82 or Table 4-83. (Default = xxx)			Timing register load select: 0 = Timing 0 (1F0-1F1h accept Timing 0 values) 1 = Timing 1 (1F0-1F1h accept Timing 1 values)
(1) Both Timing 0 and Timing 1 sets have common address setup and DRDY delay times as programmed in 1F6h[5:2].							

Setting up independent timing for drives is a two step process. The first step is to load the timing information sets (common, Timing 0, Timing 1). The second step is to associate each drive with one of the preloaded timing sets. Figure 4-31 is a flow chart that describes how to program the drives of the primary channel of the IDE interface with independent timing requirements, for the configuration recommended in Table 4-81. For the secondary channel, a similar procedure can be followed by changing all the indexes from 1Fxh to 17xh, and programming PCIIDE 43h[7:4].

Figure 4-31 IDE Interface Primary Channel Programming Flow Chart

Enter the IDE I/O Registers Programming Mode: Two consecutive (any other I/O cycle between these two reads will disable access to the IDE registers) 16-bit I/O reads from 1F1h followed by an 8-bit I/O write to 1F2h with a value of 03h.

↓

Set the values for Timing 0:
1) Write 0 to 1F6[0] to load Timing 0 parameters.
2) ***First*** set proper values in registers 1F0h and 1F1h for read/write pulse width and recovery times, and ***then*** program PCIIDE 43h[1:0], according to Table 4-82 or Table 4-83. (Program PCIIDE 43h[5:4] for Drive 0 of secondary channel).

↓

Set the values for Timing 1:
1) Write 1 to 1F6[0] to load Timing 1 parameters.
2) ***First*** set proper values in registers 1F0h and 1F1h for read/write pulse width and recovery times, and ***then*** program PCIIDE 43h[3:2], according to Table 4-82 or Table 4-83. (Program PCIIDE 43h[7:6] for Drive 1 of secondary channel).

↓

Program the Address Setup Time and DRDY Delay Time:
1) Follow Table 4-82 or Table 4-83 to set proper values into registers 1F6h[5:4]. Reset 1F6h[3:1] to 0.
2) The address setup time and DRDY delay time are common to both Timing 0 and Timing 1. If they are not the same mode, program the slower timings for the address setup time.

↓

Associate the drives with Common timings, Timing 0 parameters, or Timing 1 parameters: Follow Table 4-81 to set proper values in the registers 1F3h[7,3,2]. This should be done after all the read/write pulse and recovery, address setup, and DRDY delay have been set.

↓

Exit the IDE I/O Registers Programming Mode: An 8-bit I/O write to 1F2h with a value of 83h.

Loading Timing 0 and Timing 1 sets:
The parameters for Timing 0 are programmed by first setting IDE I/O 1F6h[0] = 0, followed by initializing IDE I/O 1F0h and 1F1h with the first set of read/write pulse widths and read/write recovery times. The parameters for Timing 1 are programmed by first setting IDE I/O 1F6h[0] = 1, followed by initializing IDE I/O 1F0h and 1F1h with a second set of read/write pulse widths and read/write recovery times.

Address setup time and channel ready hold time (DRDY) are common to both timing sets and are programmed in 1F6h[5:4] and 1F6h[3:1] respectively. Refer to Table 4-82 and Table 4-83 for information regarding the values that need to be programmed in the timing registers for different modes.

Associating drives with timing sets:
Each enabled drive can be associated with one of the preloaded timings according to Table 4-81, by programming IDE I/O registers 1F3h[7] and 1F3h[3:2] (these ***must*** be programmed after setting up the timing sets. For every enabled drive, the IDE controller allows two basic choices for timing control, “Basic”, or “Enhanced”, depending on the value of IDE I/O register 1F3h[7].

“Basic” choices (IDE I/O 1F3h[7] = 0):

1. The “common” timings as described in Section 4.11.4.1, "Enabling Common Timing for All Drives", where timing for all drives is determined by PCIIDE 40h[1:0] (Modes 0-3), or PCIIDE 43h[7:0] (Modes 4-5).
2. Timing 0

“Enhanced” choices (IDE I/O 1F3h[7] = 1):

1. Timing 0.
2. Timing 1.

Table 4-82 and Table 4-83 show the timing and recommended register settings for various IDE modes defined in the Enhanced IDE Specifications. They include PIO transfer, Single-Word DMA transfer, and Multi-Word DMA transfer modes. The actual cycle time equals the sum of actual command active time and actual command inactive (command recovery and address setup) time. These three timing requirements should be met. In some cases, the minimum cycle time requirement is greater than the sum of the command pulse and command recovery time. This means either the command active (command pulse) or command inactive time (command recovery and address setup) can be lengthened to ensure that the minimum cycle times are met.

Table 4-81 Independent Timing Selection Options for Primary Channel

Drive 1 Timing	Drive 0 Timing	1F3h[7]	1F3h[3]	1F3h[2]
Common(1)	Common(1)	0	0	0
Common(1)	Timing 0	0	0	1
Timing 0	Common(1)	0	1	0
Timing 0	Timing 0	0	1	1
Timing 1	Timing 1	1	0	0
Timing 1(2)	Timing 0	1	0	1
Timing 0	Timing 1	1	1	0
Timing 0	Timing 0	1	1	1

(1) Refer to PCIIDE 40h[1:0] for common timing values if PCIDV1 4Fh[6] = 1.
(2) Recommended configuration.

Table 4-82 16-Bit Timing Parameters with 33MHz PCI Bus

Parameter: Register Bits	Dimension	IDE Transfer Modes											
		PIO Modes						Multi-Word DMA Modes			Single-Word DMA Modes		
		0	1	2	3	4	5	0	1	2	0	1	2
R/W Command Pulse: 1F0h/170h/1F1h/ 171h[7:4], Index-0/1	Bit values in hex	5	4	3	2	2	2	7	2	2	F	8	4
	Timing in PCICLKs(1)	6	5	4	3	3	3	8	3	3	16	9	5
	Enhanced IDE Spec in ns(2)	165	125	100	80	70	N/S	215	80	70	480	240	120
R/W Recovery Time: 1F0h/170h/1F1h/ 171h[3:0], Index-0/1	Bit values in hex	9	4	0	0	0	0	6	0	0	D	4	0
	Timing in PCICLKs(1)	11	6	2	1	0	0	8	1	0	15	6	2
	Enhanced IDE Spec in ns(2)	N/S	N/S	N/S	70	25	N/S	215	50	25	N/S	N/S	N/S
Address Setup: 1F6h/176h[5:4]	Bit values in hex	2	1	1	1	0	0	0	0	0	0	0	0
	Timing in PCICLKs(1)	3	2	2	2	1	1	1	1	1	1	1	1
	Enhanced IDE Spec in ns(2)	70	50	30	30	25	N/S	N/A	N/A	N/A	N/A	N/A	N/A
DRDY: 1F6h/176h[3:1]	Bit values in hex	0	0	0	0	0	0	0	0	0	0	0	0
	Timing in PCICLKs(1)	2	2	2	2	2	2	2	2	2	2	2	2
Enhanced Mode:(3) PCIIDE 43h bits [7:6], [5:4], [3:2], or [1:0]	Bit values in hex	0	0	0	1	2	2	0	1	2	0	0	0
Cycle Time	Timing in PCICLKs	20	13	8	6	5	4	17	5	4	32	16	8
	Enhanced IDE Spec in ns(2)	600	383	240	180	120	N/S	480	150	120	960	480	240

N/S = Not Specified, N/A = Not Applicable
(1) The actual timing (in PCICLKs) that will be generated by the IDE controller if the recommended bit values in hex are programmed.
(2) The timing (in ns) as specified in the Enhanced IDE Specification.
(3) PCIIDE 43h can be programmed only after the R/W command pulse, and R/W recovery times are programmed.

Table 4-83 16-Bit Timing Parameters with 25MHz PCI Bus

Parameter: Register Bits	Dimension	IDE Transfer Modes											
		PIO Modes						Multi-Word DMA Modes			Single-Word DMA Modes		
		0	1	2	3	4	5	0	1	2	0	1	2
R/W Command Pulse: 1F0h/170h/1F1h/171h[7:4], Index-0/1	Bit values in hex	4	3	2	2	1	1	5	2	1	D	6	3
	Timing in PCICLKs(1)	5	4	3	3	2	2	6	3	2	13	7	4
	Enhanced IDE Spec in ns(2)	165	125	100	80	70	N/S	215	80	70	480	240	120
R/W Recovery Time: 1F0h/170h/1F1h/171h[3:0], Index-0/1	Bit values in hex	6	2	0	0	0	0	4	0	0	8	2	0
	Timing in PCICLKs(1)	8	4	2	1	0	0	6	1	0	10	4	1
	Enhanced IDE Spec in ns(2)	N/S	N/S	N/S	70	25	N/S	215	50	25	N/S	N/S	N/S
Address Setup: 1F6h/176h[5:4]	Bit values in hex	1	1	0	0	0	0	0	0	0	0	0	0
	Timing in PCICLKs(1)	2	2	1	1	1	1	1	1	1	1	1	1
	Enhanced IDE Spec in ns(2)	70	50	30	30	25	N/S	N/A	N/A	N/A	N/A	N/A	N/A
DRDY: 1F6h/176h[3:1]	Bit values in hex	0	0	0	0	0	0	0	0	0	0	0	0
	Timing in PCICLKs(1)	2	2	2	2	2	2	2	2	2	2	2	2
Enhanced Mode:(3) PCIIDE 43h bits [7:6], [5:4], [3:2], or [1:0]	Bit values in hex	0	0	0	1	2	2	0	1	2	0	0	1
Cycle Time	Timing in PCICLKs	15	10	6	5	4	3	13	4	3	24	12	6
	Enhanced IDE Spec in ns(2)	600	383	240	180	120	N/S	480	150	120	960	480	240

N/S = Not Specified, N/A = Not Applicable

(1) Actual timing (in PCICLKs) that will be generated by the MIDE Module if the recommended bit values in hex are programmed.

(2) Timing (in ns) as specified in the Enhanced IDE Specification.

(3) PCIIDE 43h can be programmed only after the R/W command pulse, and R/W recovery times are programmed.

4.11.4.3 Programming and Drive Placement Tips:

1. Ensure that IDE I/O Register 1F6[0] (176h[0] in the secondary channel) is set to 0 whenever accessing Timing Set 0. It is a common mistake that after accessing Timing Set 0, this bit is not reset to 0 by the BIOS. These bits will not be reset during a soft reset. After a soft reset, if the BIOS reloads Timing 0 and Timing 1 Sets, it would actually load the Timing 1 Set twice.

2. The address setup and recovery time are shared by the two IDE devices on the same channel at 1F6h[5:1]. If these two devices are not in the same mode, slower address setup and recovery time should be programmed to ensure proper timings on the slower drive. Under this assumption, two drives should be placed on the separate channels in a two-drive system. In a multiple-drive system, place slower drives on one channel and faster drives on the other channel.

3. If no IDE hard drives are in the primary slave, secondary master location or slave location, set only the command pulse and recovery time (1F0h/1F1h, Index-1, 170h/171h, Index-0 and 170h/171h, Index-1) to correspond to PIO Mode 0. This is to ensure proper timing for an ATAPI CD-ROM that may be in any of these locations.

4. If no device is present in the primary slave (Drive 1) location, set the command pulse and recovery time (1F0h and 1F1h, Timing 0 to correspond to PIO Mode 0. Also, if no drive is present in the secondary master (Drive 0), or secondary slave (Drive 1) location, set the command pulse and recovery time (170h-171h for Timing 0), and (170h-171h for Timing 1) to correspond to Mode 0. These registers do not default to a fixed value.

4.11.5 Bus Mastering Support Overview

FireStar provides a full function PCI local bus IDE controller capable of programmed I/O (PIO) mode or master mode operation. The chipset is capable of arbitrating for the ownership of the PCI local bus and transfers data between the IDE device and local memory. The IDE controller in FireStar conforms to the ATA Standard for IDE disk controllers.

By performing the IDE data transfers as a bus master instead of a slave, the chipset off-loads the CPU from having to perform the transfers. This benefit is realized in the form of the CPU not having to perform programmed I/O transfers to effect the data transfer between the disk and the memory.

Figure 4-32 shows the configuration for a bus mastering IDE controller. Figures 4-33 and 4-34 depict the timing parameters associated with Multi-Word and Single-Word DMA transfers.

The master mode of operation is an extension to the standard IDE controller model. Thus, systems can still revert back to slave mode IDE if they so desire. The master mode of operation is designed to work with any IDE device that supports DMA transfers on the IDE bus. Devices that do not support DMA on the IDE bus can transfer IDE data using programmed I/O.

Figure 4-32 Master IDE Configuration

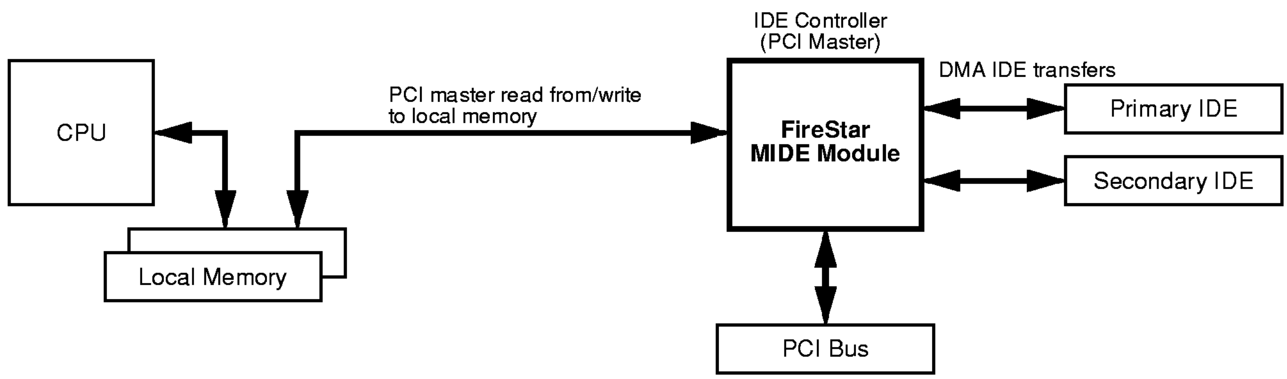


Figure 4-33 Multi-Word DMA Transfer Mode

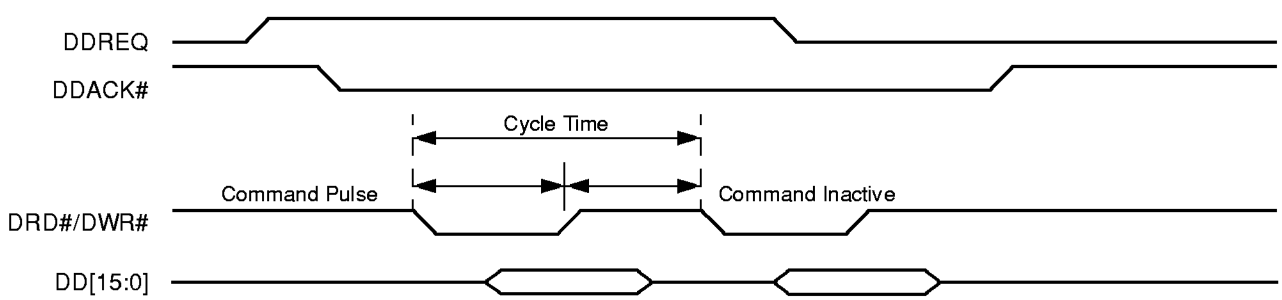
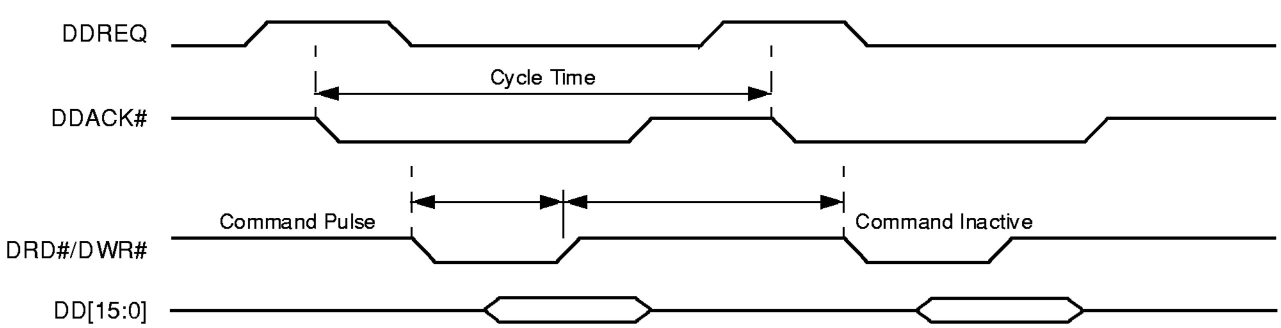


Figure 4-34 Single-Word DMA Transfer Mode



4.11.6 Physical Region Descriptor Table

Before the IDE controller starts a master transfer it is given a pointer to a Physical Region Descriptor Table. This table contains some number of Physical Region Descriptors (PRDs) which describe areas of memory that are involved in the data transfer. The descriptor table must be aligned on a 4-byte boundary and the table cannot cross a 64K boundary in memory.

4.11.6.1 Physical Region Descriptor

The physical memory region to be transferred is described by a Physical Region Descriptor (PRD). The data transfer will proceed until all the regions described by the PRDs in the table have been transferred. The format of a PRD table is shown in Table 4-84.

Each Physical Region Descriptor entry is eight bytes in length:

- The first four bytes specify the start address of a physical memory region.
- The next two bytes specify the size of the region in bytes (64K byte limit per region). A value of zero in these two bytes indicates 64K.
- Bit 7 of the last byte indicates the end of the table; bus master operation terminates when the end of the table has been reached.

Refer to Figure 4-35 for the correlation between PRD table entries and memory regions that are involved in DMA data transfers.

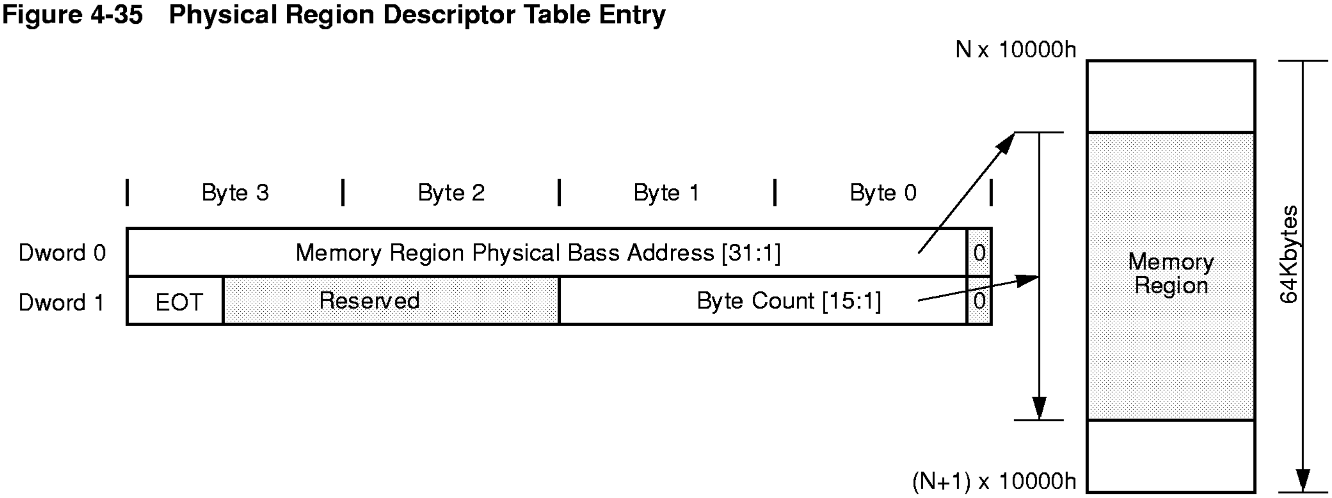
The memory region specified by the PRD cannot straddle a 64Kbyte boundary. Also, the total sum of the PRD byte counts must be equal to or greater than the size of the disk transfer request.

Table 4-84 Physical Region Descriptor Table Entry

Bit(s)	Name	Default	Function
Byte-0, bit 0	---	0	0 (RO)
Byte-[3:1] Byte-0, bits [7:1]	BASE	xxxx xxxx	Memory Region Physical Base Address [31:1]
Byte-4, bit 0	---	0	0 (RO)
Byte-5 Byte-4, bits [7:1]	COUNT	xxxx	Byte Count [15:1]
Byte-6	---	xx	Reserved
Byte-7, bits [6:0]	---	xx	Reserved
Byte-7, bit 7	EOT	x	End of Table

Note: The memory region specified by the descriptor is further restricted such that the region cannot straddle a 64K boundary. This means the byte count is limited to 64K and the incrementer for the current address register only extends from bit 1 to bit 15.

Figure 4-35 Physical Region Descriptor Table Entry



4.11.6.2 Bus Master IDE Registers

The bus master IDE function uses 16 bytes of I/O space. The base address of this block of I/O space is pointed to by the Bus Master IDE Base Address Register and PCIIDE 20h-23h. All bus master IDE I/O space registers can be accessed as byte, word, or dword quantities. The description of the 16 bytes of I/O registers is shown in Table 4-85 (refer to Section 5.4.3, "Bus Master IDE Registers" for individual bit formats in each register).

Table 4-85 Bus Master IDE Registers

Offset from Base Address	Register Access	Register Name/Function
00h	R/W	Bus Master IDE Command Register for Primary IDE
01h		Device-specific
02h	RWC	Bus Master IDE Status Register for Primary IDE
03h		Device-specific
04h-07h	R/W	Bus Master IDE PRD Table Address for Primary IDE
08h	R/W	Bus Master IDE Command Register for Secondary IDE
09h		Device-specific
0Ah	RWC	Bus Master IDE status Register for Secondary IDE
0Bh		Device-specific
0Ch-0Fh	R/W	Bus Master IDE PRD Table Address for Secondary IDE

4.11.6.3 Standard Programming Sequence for Bus Mastering Operations

DMA Mode capability can be programmed independently for primary and secondary channels by setting the bus mastering registers 02h and 0Ah. Ensure that PIO Mode 3 is set up in PCIIDE 40h[1:0], as described in Section 4.11.4.1, "Enabling Common Timing for All Drives". Subsequently, the appropriate bits in PCIIDE 43h[7:0] can be set to enable DMA Mode 1 or DMA Mode 2.

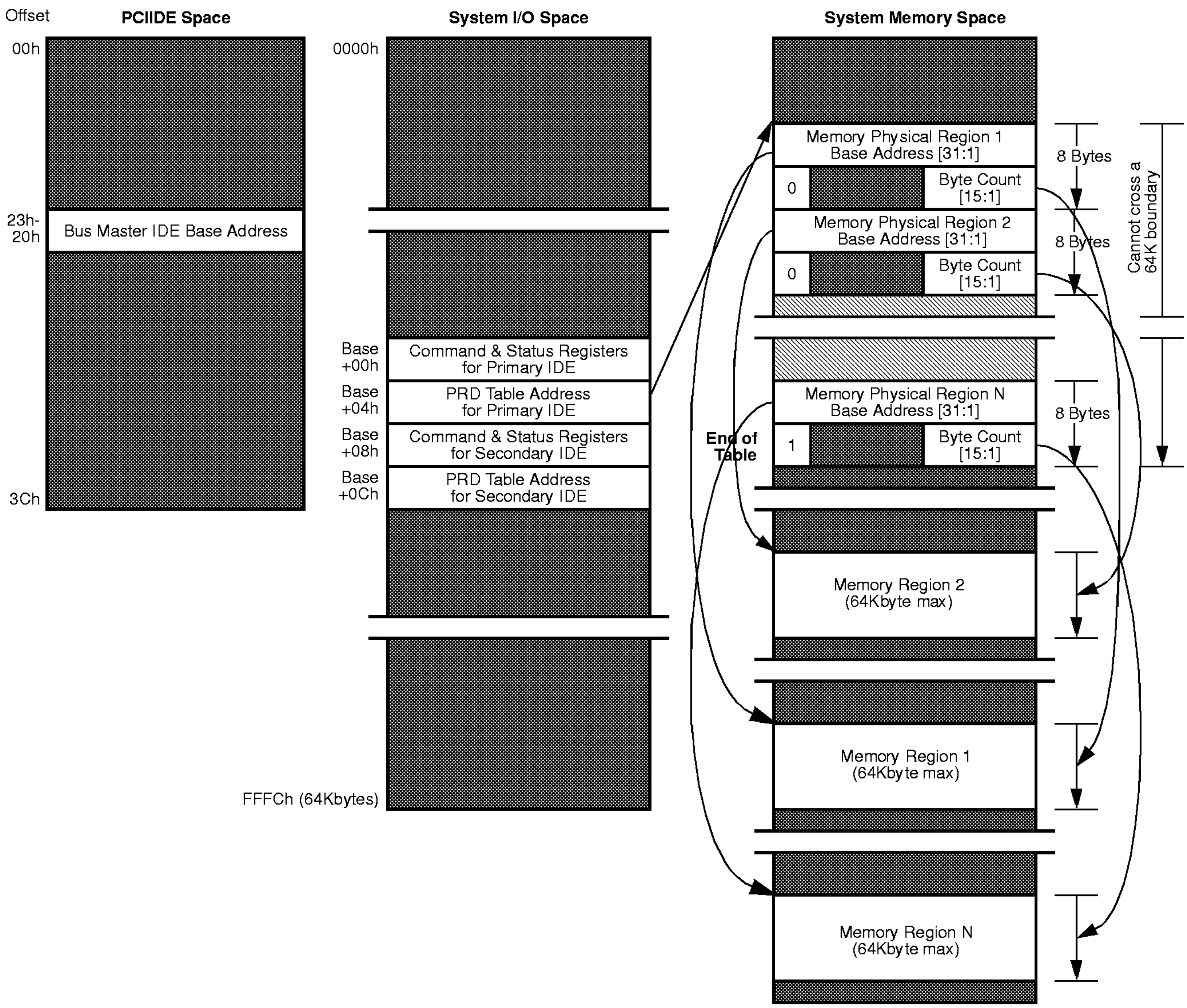
Table 4-86 DMA Mode Programming Bits

7	6	5	4	3	2	1	0
Base Address + 02h		**Bus Master IDE Status Register for Primary IDE**					**Default = 00h**
	Drive 1 DMA capable: This bit is set by device-dependent code (BIOS or device driver) to indicate that Drive 1 for this channel is capable of DMA transfers, and that the controller has been initialized for optimum performance.	Drive 0 DMA capable: This bit is set by device-dependent code (BIOS or device driver) to indicate that Drive 0 for this channel is capable of DMA transfers, and that the controller has been initialized for optimum performance.					
Base Address + 0Ah		**Bus Master IDE Status Register for Secondary IDE**					**Default = 00h**
	Drive 1 DMA Capable: This bit is set by device dependent code (BIOS or device driver) to indicate that Drive 1 for this channel is capable of DMA transfers, and that the controller has been initialized for optimum performance.	Drive 0 DMA Capable: This bit is set by device dependent code (BIOS or device driver) to indicate that Drive 0 for this channel is capable of DMA transfers and that the controller has been initialized for optimum performance.					

To initiate a bus master transfer between memory and an IDE DMA slave device, the following steps are required (Figure 4-36 shows the bus master operations described below):

1. DOS calls BIOS (INT13H) to start a disk transfer. For OS that is not using BIOS services, a device driver should be in place to intercept the disk access request.
2. BIOS (or device driver) prepares a PRD Table in the system memory. Each PRD is 8 bytes long. It consists of an address pointer (the location specifies in the ES:BXh) to the starting address and the transfer count (the sector count specifies in the AL) of the memory buffer to be transferred. If the data area (as pointed by ES:BXh and AL) crosses a 64K boundary, the BIOS (or device driver) would have to break it into multiple transfers (PRDs) so that each of them lies within the boundary.
3. BIOS (or device driver) provides the starting address of the PRD Table by loading the PRD Table Pointer Register (Bus Master IDE Base Address + 04h or + 0Ch).
4. The direction of the data transfer is specified by setting the Read/write Control bit in the Command Register (Bus Master IDE Base Address + 00h or + 08h). Clear the Interrupt bit and Error bit in the Status Register (Bus Master IDE Base Address + 02h or + 0Ah).
5. BIOS (device driver) resets the Hard Disk Task Complete Flag (a memory byte location at 40:E8h) to 00h. It will be in a tight loop checking whether this Complete Flag is set to FFh. Other OS may have a different mechanism to detect disk activity.
6. BIOS (or device driver) specifies address and size of the data request by programming the Command Block Registers of the IDE device and issues the appropriate DMA transfer command to it.
7. BIOS (device driver) engages the bus master function by writing ‘1’ to the Start bit in the Command Register (Bus Master IDE Base Address + 00h and + 08h) for the appropriate channel.
8. The 82C700 starts reading the first PRD and transfers data to/from its 32-byte FIFO in response to DMA requests (IDEREQx) from the IDE devices. The 82C700 then starts transfer to local memory (an internal master request will be generated) for read if the FIFO is empty, for write if the FIFO is full, or when the byte count expires and no more entries in the PRD.
9. After the last data transfer within a PRD, the 82C700 checks the End of Table (EOT) bit to decide whether to read another PRD or move forward.
10. At the end of transfer, the IDE device signals an interrupt.
11. After 82C700 has flushed all the data from its FIFO to system memory, it resets the Bus Master IDE Active bit and sets the Interrupt bit in the Status Register.
12. The disk interrupt (DINTx) will be passed to the internal INTC (‘8259). DINTx will not be blocked to the INTC in a disk write operation. This DINTx triggers IRQ14 or IRQ15 and eventually goes to interrupt the CPU.
13. CPU generates an INTA# cycle in responds to the IRQ14 (INT76H) or IRQ15 (INT77H). The INT76H handler sets 40:E8h to FFh to signal completion of the disk access.
14. BIOS (device driver) resets the Start/Stop bit in the Command Register. It then reads the Status and Interrupt bits in the Status Register and then the IDE device’s status to determine if the transfer completed successfully.
15. Status will be passed back to INT13H to finish the operation.

Figure 4-36 Bus Master IDE Operation



4.11.6.4 Programming the IDE Interrupt Routing

Table 4-87 details the interrupt routing mechanism for the MIDE Module while in the Legacy and Native Modes. The system BIOS needs to program them accordingly.

Table 4-87 IDE Interrupt Routing Chart

Functions		PCI IDE Configuration Register Setting				IDE Drive Interrupts routed to:	
IDE Modes		04h[0]	40h[3]	40h[2]	09h[3:0]	Primary	Secondary
Primary	Secondary	IDE I/O Enable	2nd IDE Disable	Native Mode Enable	Native/ Legacy Mode	8259 Interrupt or PCI Interrupt	8259 Interrupt or PCI Interrupt
Disabled		0	x	x	xxxx	N/A	N/A
Legacy[1]	Disabled	1	1	0	xxxx	8259 IRQ14 input	N/A
		1	1	1	xx10		
Native	Disabled	1	1	1	xx11	PCIRQ3#[2]	N/A
Legacy[1]	Native	1	0	1	1110	8259 IRQ14 input	PCIRQ3#[2]
Native	Legacy[1]	1	0	1	1011	PCIRQ3#[2]	8259 IRQ15 input
Legacy[1]	Legacy[1]	1	0	0	xxxx	8259 IRQ14 input	8259 IRQ15 input
		1	0	1	1010		
Native	Native	1	0	1	1111	PCIRQ3#[2]	PCIRQ3#[2]

(1) The 8259 interrupt input IRQ14 (IRQ15) will not be available for mapping from PCIIRQ0#-3# if the primary (secondary) channel is enabled in legacy mode.

(2) See Table 4-88 for selection possibilities.

Table 4-88 IDE Interrupt Selection Registers

7	6	5	4	3	2	1	0
PCIIDE 45h		**IDE Interrupt Selection Register**					**Default = 00h**
Secondary Drive 1 interrupt pin: 00 = IRQ10+PCIRQ0# 01 = IRQ11+PCIRQ1# 10 = IRQ14+PCIRQ2# 11 = IRQ15+PCIRQ3#		Secondary Drive 0 interrupt pin: 00 = IRQ10+PCIRQ0# 01 = IRQ11+PCIRQ1# 10 = IRQ14+PCIRQ2# 11 = IRQ15+PCIRQ3#		Primary Drive 1 interrupt pin: 00 = IRQ10+PCIRQ0# 01 = IRQ11+PCIRQ1# 10 = IRQ14+PCIRQ2# 11 = IRQ15+PCIRQ3#		Primary Drive 0 interrupt pin: 00 = IRQ10+PCIRQ0# 01 = IRQ11+PCIRQ1# 10 = IRQ14+PCIRQ2# 11 = IRQ15+PCIRQ3#	
Note: ISA IRQ is selected for Legacy Mode and PCI IRQ is selected for Native Mode (see PCIIDE 09h).							
PCIIDE 47h - FS ACPI Version		**IDE Interrupt Selection Register**					**Default = FAh**
Secondary Drive 1 interrupt pin: 00 = IRQ10+PCIRQ0# 01 = IRQ11+PCIRQ1# 10 = IRQ14+PCIRQ2# 11 = IRQ15+PCIRQ3#		Secondary Drive 0 interrupt pin: 00 = IRQ10+PCIRQ0# 01 = IRQ11+PCIRQ1# 10 = IRQ14+PCIRQ2# 11 = IRQ15+PCIRQ3#		Primary Drive 1 interrupt pin: 00 = IRQ10+PCIRQ0# 01 = IRQ11+PCIRQ1# 10 = IRQ14+PCIRQ2# 11 = IRQ15+PCIRQ3#		Primary Drive 0 interrupt pin: 00 = IRQ10+PCIRQ0# 01 = IRQ11+PCIRQ1# 10 = IRQ14+PCIRQ2# 11 = IRQ15+PCIRQ3#	
Note: ISA IRQ is selected for Legacy Mode and PCI IRQ is selected for Native Mode (see PCIIDE 09h).							

4.11.7 Emulated Bus Mastering Mode

FireStar provides a means to have DMA-type operation on a PIO mode drive. Any IDE device can use this feature. The feature works by acting as a bus master, in much the same way as the DMA controller would.

Emulated bus mastering allows devices such as IDE CD-ROM drives to access data and store it in a memory buffer with no CPU intervention. The interrupt from the IDE drive is handled by a hardware sequencer; the CPU is interrupted only after the transfer is complete. The attached IDE drive can deassert IOCHRDY as needed if data is not ready.

Since the purpose of the Emulated Bus Master IDE mode is primarily to increase system performance during the disk read from CD-ROM transfer, this mode is implemented for the read-from-disk direction only.

The differences between PIO mode and DMA mode transfer are not only the control signal behavior, but also the programming methods to the drive and IDE controller. These are described below. It is important to note that **no DMA instructions are allowed** to be sent to the drive in this mode, since the drive itself is a PIO-mode-only device. No interception of drive-related commands takes place, so DMA-related commands would only confuse the IDE drive logic.

4.11.7.1 Setup

The IDE controller channel must be programmed to support a bus master IDE drive. In addition, the bit corresponding to the channel must be set in PCIIDE 44h[3:0] for FireStar and PCI-IDE 46h[3:0] for FireStar ACPI to select Emulated Bus Mastering (refer to Table 4-89).

4.11.7.2 Operation

As stated earlier, only PIO-mode commands are allowed to the drive. The emulated bus mastering function automates only reads from the drive, so writes must still be carried out through normal I/O write cycles from the CPU.

Emulated bus mastering depends on the IRQ line to determine transfer completion. For CD-ROM data read, IRQ is asserted for two reasons:

- When data is ready to be read by host (“data ready IRQ”)
- At the end of the transfer (“transfer complete IRQ”).

This behavior is important to understand for the emulation implementation.

Data Ready IRQ: When IRQ assertion signals “data ready”, software must act as if it is programming a DMA mode IDE controller by doing the following:

- Prepare the required PRD table and word count in memory
- Generate required control commands to the IDE controller. Since the drive used is not DMA-capable, software must not send any DMA commands to the disk drive. Only normal PIO commands can be used.
- Set the Start bit in the IDE controller register.

At this point the hardware takes over and generates the programmed number of I/O read cycles to the CD-ROM drive. During the transfer, the IDE controller uses the existing bus master IDE control state machine and logic to perform the operation. The signal DCS1# is forced active, DCS3# is forced inactive, and DDACK# is masked, before being sent to the disk drive interface. Therefore, the CD-ROM is seeing PIO mode data read transfer control.

Transfer Complete IRQ: When the word count expires in the IDE controller and CD-ROM drive, IRQ is asserted to signal “transfer completed”. Software must:

- Reset the Start bit in the IDE controller register
- Read the controller status and then the drive status to determine whether the transfer completed successfully.

Table 4-89 Emulated Bus Master Control Registers

7	6	5	4	3	2	1	0
PCIIDE 44h		**Emulated Bus Master Register**					**Default = 00h**
Reserved				Emulated bus mastering for Cable 1, Drive 1: 0 = Disable 1 = Enable	Emulated bus mastering for Cable 1, Drive 0: 0 = Disable 1 = Enable	Emulated bus mastering for Cable 0, Drive 1: 0 = Disable 1 = Enable	Emulated bus mastering for Cable 0, Drive 0: 0 = Disable 1 = Enable
PCIIDE 46h - FS ACPI Version		**Emulated IDE Configuration Register**					**Default = 00h**
Fix for I/O 32-bit Mode 4 and Mode 5 timing: 0 = Disable 1 = Enable	Reserved			Emulated bus mastering for Cable 1, Drive 1: 0 = Disable 1 = Enable	Emulated bus mastering for Cable 1, Drive 0: 0 = Disable 1 = Enable	Emulated bus mastering for Cable 0, Drive 1: 0 = Disable 1 = Enable	Emulated bus mastering for Cable 0, Drive 0: 0 = Disable 1 = Enable

4.12 Type F DMA Support

Improved DMA transfer performance is available on a channel-by-channel basis for those devices capable of shorter ISA command pulses. Normally, the FireStar DMA cycle width is six ISA clocks for the read command and four ISA clocks for the write command. Enabling Type F DMA for a channel changes this timing.

- For ISA DMA devices:
  - Read command (IOR# or MRD#) is two ISA clocks
  - Write command (IOW# or MWR#) is one ISA clock
- For CISA DMA devices:
  - CMD# is three ISA clocks
  - MRD# or MWR# is also three ISA clocks

Type F DMA is controlled through the EISA register scheme. Only the bits shown in Table 4-90 are supported.

Table 4-90 Type DMA Control Register Bits

7	6	5	4	3	2	1	0
Port 40Bh		**EISA DMA Extended Mode Register, Channels 0 through 3**					**Default = xxh**
Not implemented	Not implemented	Cycle timing: 00 = ISA-compatible 01 = ISA-compatible 10 = ISA-compatible 11 = Type F		Not implemented	Not implemented	DMA Channel: 00 = Channel 0 01 = Channel 1 10 = Channel 2 11 = Channel 3	
Port 4D6h		**EISA DMA Extended Mode Register, Channels 4 through 7**					**Default = xxh**
Not implemented	Not implemented	Cycle timing: 00 = ISA-compatible 01 = ISA-compatible 10 = ISA-compatible 11 = Type F		Not implemented	Not implemented	DMA Channel: 00 = Reserved 01 = Channel 5 10 = Channel 6 11 = Channel 7	

4.13 Distributed DMA Overview

FireStar incorporates features to support Distributed DMA. The following subsections describe the Distributed DMA protocol and how FireStar supports it.

4.13.1 Distributed DMA Protocol

DMA on a PCI bus or across a PCI bridge is not currently handled by either the PCI or CardBus specifications. To fill this need, a DMA protocol has been developed. The protocol provides a solid framework for compatible operation, but does not specify the exact method of implementation. Therefore, this document describes the generally agreed-to protocol and highlights its implementation in OPTi designs.

4.13.1.1 Introduction

The distributed DMA protocol allows PCI-based designs to incorporate multiple DMA controller (DMAC) channels distributed throughout the system, each of which is local to the device it will service. The PCI specification itself is not modified for DMA since only standard I/O and memory cycles are used in this scheme.

A specific protocol is needed for multiple DMA controllers on PCI. If each DMA channel had its own unique set of registers, there would be no problem; the device responsible for each channel would claim only its own accesses. Unfortunately, in the PC architecture some DMA registers are shared by groups of four channels; up to four separate devices would have to claim a single I/O read access, with disastrous results.

Therefore, the DMAC protocol specifies the means of:

- Claiming and routing I/O accesses to the correct owner of each channel
- Dividing up accesses that could be claimed by multiple devices
- Returning combined status information from multiple sources.

The means by which the distributed DMA protocol defines these responsibilities is described next.

4.13.1.2 Protocol Overview

The basic protocol simply defines new and unique I/O addresses for each register on every DMAC channel. The remapping puts all registers associated with a specific DMAC channel into a 10h byte area to make windowing requirements easier on PCI-to-PCI bridges.

When DMAC channels are present on a remote bus, the PCI controller sends DMA register I/O read and write cycles to the local PCI bus PCI-to-PCI bridges that connect the remote DMAC channels. PCI-to-PCI bridges need not be DMA-aware to pass these cycles, as long as they have an I/O mapping window programmed to claim the remapped accesses.

4.13.1.3 Distributed DMA Protocol Terminology

Devices on PCI that adhere to the distributed DMA protocol are referred to in this document using the phrases "master DMAC", "DMA Channel Selector Register", "remote DMAC channels", and "DMA remapper". These terms are described below.

Master DMAC

There must be one master DMAC in the system. It is an OPTi standard 82C206-type DMAC subsystem with shadow register provisions. The master DMAC:

- Becomes the claimer of cycles to DMAC channels that are not used by PCI peripheral devices or devices on the secondary side of PCI-to-PCI or PCI-to-ISA bridges.
- Provide all seven DMA channels: in the event that no other devices in the system support DMA, the master DMAC must claim all cycles.
- Claims all accesses for DMA Channel 4.

Remote DMAC Channels

Remote DMAC channels can be anywhere in the system, even on the same PCI bus as the Master DMAC. Each remote DMAC channel must claim only the remapped cycles for which it is responsible. The only other difference between a remote DMAC channel and a channel on the master DMAC is that the master DMAC shadows writes to be able to respond to reads of shadowed information. Remote DMAC channels never respond to reads for write-only registers in the 8237 design.

DMA Channel Selector Register

Within the PCI Configuration Registers of PCI-based DMACs and DMA-aware PCI-to-PCI bridges are seven configuration bits to select whether each DMA channel is local or remote. For each device, the bits are programmed to select whether the DMAC claims that DMA channel or not. "Claimed" means that the channel is claimed by the device or that the device is claiming the cycle on behalf of another device downstream. For the scheme to work properly, each channel can be assigned "claimed" status in only one DMA Channel Selector Register; any channels that are unclaimed should be assigned to the master DMAC.

DMAC Responsibility - This bit determines whether the concerned DMAC will be the system master. Only one master is possible in the system.

The DMA Channel Selector Register layout is illustrated in Table 4-91.

After master and remote status has been properly assigned, the responsibility for claiming cycles can be defined as discussed next.

DMA Remapper

The address of each DMA controller port for each channel is normally listed as an absolute value in the ISA-compatible I/O address space. The DMA remapper remaps these ports through a lookup table scheme. For the most part, the assignments are regular enough that a formula could be applied. Unfortunately, certain ISA-compatible register locations (the Page Register in particular) introduce an irregularity in the remapping and require an inconsistent approach. The mapping is illustrated in Table 4-92 using DMA Channel 0 as an example.

From the CPU instruction set point of view, no change in addressing is required. All code can continue to issue the original ISA-compatible port addresses. However, DMA programming code that is PCI-aware can directly address these ports if desired.

Note that only the EISA extensions to the Page Register and the Count Register are implemented. The remaining EISA extensions are not currently handled by this protocol.

Table 4-91 DMA Channel Selector Register

7	6	5	4	3	2	1	0
PCIDV1 5Ch			**DMA Channel Selector Register**				**Default = 00h**
Ch 7 (DMAC2): 0 = Local 1 = On PCI	Ch 6 (DMAC2): 0 = Local 1 = On PCI	Ch 5 (DMAC2): 0 = Local 1 = On PCI	Hardware Distributed DMA: 0 = Disable 1 = Enable	Ch 3 (DMAC1): 0 = Local 1 = On PCI	Ch 2 (DMAC1): 0 = Local 1 = On PCI	Ch 1 (DMAC1): 0 = Local 1 = On PCI	Ch 0 (DMAC1): 0 = Local 1 = On PCI

Table 4-92 DMA Remap Scheme - Generic for all DMA Channels

Register	Bits	Type	ISA I/O Address Example: Channel 0	Remapped Offset for PCI
Memory Address w/byte ptr low	A[7:0]	Read/Write	000h	b+(ch*10)+000h
Memory Address w/byte ptr high	A[15:8]	Read/Write	000h	b+(ch*10)+001h
Page Address	A[23:16]	Read/Write	087h	b+(ch*10)+002h
EISA High Byte Page Address	A[31:24]	Read/Write	487h	b+(ch*10)+003h
Count w/byte ptr low	C[7:0]	Read/Write	001h	b+(ch*10)+004h
Count w/byte ptr high	C[15:8]	Read/Write	001h	b+(ch*10)+005h
EISA High Byte Count	C[23:16]	Read/Write	401h	b+(ch*10)+006h
Reserved			007h	--
Status		Read-Only	008h	b+(ch*10)+008h
Command		Write-Only	008h	b+(ch*10)+008h
DMA Request		Write-Only	009h	b+(ch*10)+009h
Set Single Mask Bit		Write-Only	00Ah	b+(ch*10)+00Fh[0]
Mode		Write-Only	00Bh	b+(ch*10)+00Bh
Byte Pointer Flip-Flop Clear		Write-Only	00Ch	handled by DMA remapper
Master Clear		Write-Only	00Dh	b+(ch*10)+00Dh
Mask Clear		Write-Only	00Eh	b+(ch*10)+00Fh[0]
Mask		Read/Write	00Fh	b+(ch*10)+00Fh[0]

Notes:

'b' indicates base address

'ch' indicates channel number: ch=0 for channel 0, ch=1 for channel 1, ch=2 for channel 2, …, ch=7 for channel 7

Table 4-93 Complete Remap Scheme, Channels 0-3

Register	Type	ISA I/O Port Address / PCI Remapped Address			
		DMA Ch 0	DMA Ch 1	DMA Ch 2	DMA Ch 3
Memory Address w/byte ptr low	Read/Write	000h→b+000h	002h→b+010h	004h→b+020h	006h→b+030h
Memory Address w/byte ptr high	Read/Write	000h→b+001h	002h→b+011h	004h→b+021h	006h→b+031h
Page Address	Read/Write	087h→b+002h	083h→b+012h	081h→b+022h	082h→b+032h
EISA High Byte Page Address	Read/Write	487h→b+003h	483h→b+013h	481h→b+023h	482h→b+033h
Count w/byte ptr low	Read/Write	001h→b+004h	003h→b+014h	005h→b+024h	007h→b+034h
Count w/byte ptr high	Read/Write	001h→b+005h	003h→b+015h	005h→b+025h	007h→b+035h
EISA High Byte Count	Read/Write	401h→b+006h	403h→b+016h	405h→b+026h	407h→b+036h
Status	Read-Only	008h→b+008h--b+018h--b+028h--b+038h (four reads)			
Command	Write-Only	008h→b+008h--b+018h--b+028h--b+038h (four writes)			
DMA Request	Write-Only	009h→b+009h	009h→b+019h	009h→b+029h	009h→b+039h
Set Single Mask Bit	Write-Only	00Ah→b+00Fh[0]	00Ah→b+01Fh[0]	00Ah→b+02Fh[0]	00Ah→b+03Fh[0]
Mode	Write-Only	00Bh→b+00Bh	00Bh→b+01Bh	00Bh→b+02Bh	00Bh→b+03Bh
Byte Pointer Flip-Flop Clear	Write-Only	00Ch→used by remapper, but no remapped I/O cycle generated			
Master Clear	Write-Only	00Dh/b+00Dh--b+01Dh--b+02Dh--b+03Dh (four writes)			
Mask Clear	Write-Only	00Eh→b+00Fh[0]--b+01Fh[0]--b+02Fh[0]--b+03Fh[0] (four writes)			
Mask	Read/Write	00Fh→b+00Fh[0]--b+01Fh[0]--b+02Fh[0]--b+03Fh[0] (four writes)			

Table 4-94 Complete Remap Scheme, Channels 4-7

Register	Type	ISA I/O Port Address / PCI Remapped Address			
		DMA Ch 4	DMA Ch 5	DMA Ch 6	DMA Ch 7
Memory Address w/byte ptr low	Read/Write	0C0h→none	0C4h→b+050h	0C8h→b+060h	0CCh→b+070h
Memory Address w/byte ptr high	Read/Write	0C0h→none	0C4h→b+051h	0C8h→b+061h	0CCh→b+071h
Page Address	Read/Write	08Fh→none	08Bh→b+052h	089h→b+062h	08Ah→b+072h
EISA High Byte Page Address	Read/Write	none→none	48Bh→b+053h	489h→b+063h	48Ah→b+073h
Count w/byte ptr low	Read/Write	0C2h→none	0C6h→b+054h	0CAh→b+064h	0CEh→b+074h
Count w/byte ptr high	Read/Write	0C2h→none	0C6h→b+055h	0CAh→b+065h	0CEh→b+075h
EISA High Byte Count	Read/Write	none→none	4C6h→b+056h	4CAh→b+066h	4CEh→b+076h
Status	Read-Only	0D0h→none	0D0h→b+058h--b+068h--b+078h (three reads)		
Command	Write-Only	0D0h→none	0D0h→b+058h--b+068h--b+078h (three writes)		
DMA Request	Write-Only	0D2h→none	0D2h→b+059h	0D2h→b+069h	0D2h→b+079h
Set Single Mask Bit	Write-Only	0D4h→none	0D4h→b+05Fh[0]	0D4h→b+06Fh[0]	0D4h→b+07Fh[0]
Mode	Write-Only	0D6h→none	0D6h→b+05Bh	0D6h→b+06Bh	0D6h→b+07Bh
Byte Pointer Flip-Flop Clear	Write-Only	0D8h→used by remapper, but no remapped I/O cycle generated			
Master Clear	Write-Only	0DAh→b+05Dh--b+06Dh--b+07Dh (three writes)			
Mask Clear	Write-Only	0DCh→none	0DCh→b+05Fh[0]--b+06Fh[0]--b+07Fh[0] (three writes)		
Mask	Read/Write	0DEh→b+05Fh[0]--b+06Fh[0]--b+07Fh[0] (three writes)			

Register Writes

Most, but not all, DMA I/O register writes are remapped by the DMA remapper. For all cases, the DMA remapper must generate STOP# in response to the original cycle until these remapped cycles are complete.

- Mode and Request
  - For these write-only DMA registers, bits [1:0] indicate the channel number. Therefore, the DMA remapper need only generate a single I/O access, to the channel specified.
- Command, Mask, and Master Clear
  - The DMA remapper remaps the access to four unique I/O locations (only three for DMAC2 accesses since DMA Channel 4 is not important). Each device claims only its own access.
- Single-Channel Mask and Mask Clear
  - These accesses simply update the Mask Register. Therefore, the DMA remapper must maintain a copy of the Mask Register internally so that it can update the mask. It then generates remapped writes to all Mask Registers.
- Byte Pointer Flip-Flop Clear
  - The DMA remapper uses this value internally to determine the remapping for address and count accesses. However, it does not generate any external I/O cycles for this write.
- All Other Registers
  - The DMA remapper remaps the I/O write according to the tables.

Register Reads

Only certain reads are remapped by the DMA remapper. Reads to other registers are reads of DMA shadow registers, which are not at industry-standard addresses and therefore are not covered by the distributed DMA protocol. Claiming DMAC register reads is straightforward. For all cases, the DMA remapper must generate STOP# in response to the original cycle until these remapped cycles are complete.

- Address, Count, and Page Address Registers
  - All reads are remapped. The channel owner claims the remapped cycle and returns the data. PCI bridges must claim this cycle and pass it on to the secondary bus to return the data.
- Mask Register
  - Reads are not remapped. The DMA remapper claims the cycle and returns shadowed information.
- Status Register
  - Reads are remapped to four unique I/O locations. The DMA remapper combines the returned status information for each channel and provides it to the requester.
- Write-only Registers
  - Reads are not remapped. The 82C206 core provides read-back capability of these registers as shadowed information.

Note that there is no provision for conflicting claims by more than one device. As long as exactly one "claimed" assignment is made for each channel, there will never be a conflict.

4.13.1.4 DMA Channel Selection

FireStar provides the feature of selectable DMA channels. The registers listed in Table 4-95 provide the selection bits.

4.13.2 Hardware Distributed DMA Support

FireStar implements a hardware DMA remapper when through PCIDV1 5Ch[4], DMA controller I/O accesses are automatically remapped to distributed DMA addresses. (Refer to Table 4-96.)

Table 4-95 Selectable DMA Channel Support

7	6	5	4	3	2	1	0
PCIDV1 C0h			**DMA Channels A and B Selection Register**				**Default = 10h**
	DMA channel selection on DRQB/DACKB# pins (Channel 1): 000 = Channel 0 001 = Channel 1 010 = Channel 2 011 = Channel 3 100 = PPWR5 101 = Channel 5 110 = Channel 6 111 = Channel 7				DMA channel selection on DRQA/DACKA# pins (Default = Channel 0): 000 = Channel 0 001 = Channel 1 010 = Channel 2 011 = Channel 3 100 = PPWR4 101 = Channel 5 110 = Channel 6 111 = Channel 7		
PCIDV1 C1h			**DMA Channels C and D Selection Register**				**Default = 32h**
	DMA channel selection on DRQD/DACKD# pins (Channel 3): 000 = Channel 0 001 = Channel 1 010 = Channel 2 011 = Channel 3 100 = PPWR7 101 = Channel 5 110 = Channel 6 111 = Channel 7				DMA channel selection on DRQC/DACKC# pins (Channel 2): 000 = Channel 0 001 = Channel 1 010 = Channel 2 011 = Channel 3 100 = PPWR6 101 = Channel 5 110 = Channel 6 111 = Channel 7		
PCIDV1 C2h			**DMA Channel E Selection Register**				**Default = 50h**
	DMA channel selection on DRQE/DACKE# pins (Default = Channel 5): 000 = Channel 0 001 = Channel 1 010 = Channel 2 011 = Channel 3 100 = PPWR13 101 = Channel 5 110 = Channel 6 111 = Channel 7						
PCIDV1 C3h			**DMA Channels F and G Selection Register**				**Default = 76h**
	DMA channel selection on DRQG/DACKG# pins (Default = Channel 7): 000 = Channel 0 001 = Channel 1 010 = Channel 2 011 = Channel 3 100 = PPWR15 101 = Channel 5 110 = Channel 6 111 = Channel 7				DMA channel selection on DRQF/DACKF# pins (Default = Channel 6): 000 = Channel 0 001 = Channel 1 010 = Channel 2 011 = Channel 3 100 = PPWR14 101 = Channel 5 110 = Channel 6 111 = Channel 7		

Table 4-96 Hardware DMA Remapper

7	6	5	4	3	2	1	0
PCIDV1 5Ch			**DMA Channel Selector Register**				**Default = 00h**
			Hardware Distributed DMA: 0 = Disable 1 = Enable				

4.13.3 Software Distributed DMA Support

The support implemented is simply the addition of I/O port monitoring. The read or write access of any of the following I/O ranges will cause a DMA_ACCESS PMI.

- 000-00Fh, 400-40Fh
- 0C0-0DFh, 4C0-4DFh
- 4E0-4FFh (EISA Stop Registers)
- 080-08Fh, 480-48Fh

Once the SMM code takes over, it can read the I/O write data from the SMM register save area of system management memory. SMM code can determine the DMA controller I/O register address that the application attempted to write by reading SYSCFG D6h, D7h, and EBh (as shown in Table 4-97).

4.13.3.1 PCI Configuration Registers

FireStar offers the PCI configuration registers shown in Table 4-98 for system event handlers to control and monitor Distributed DMA.

Table 4-97 Access Trap Address Register Bits

7	6	5	4	3	2	1	0
SYSCFG D6h			**PMU Control Register 10**				**Default = 00h**
						Access trap bit A9 (RO)	Access trap bit A8 (RO)
SYSCFG D7h			**Access Port Address Register 1**				**Default = 00h**
Access trap address bits A[7:0]: - These bits, along with SYSCFG D6h[1:0] and SYSCFG EBh[7:0] provide the 16-bit address of the port access that caused the SMI trap. - SYSCFG D6h[2] indicates whether an I/O read or an I/O write access was trapped. - SYSCFG D6h[3] gives the status of the SBHE# signal for the I/O instruction that was trapped.							
SYSCFG EBh			**Access Port Address Register 2**				**Default = 00h**
Reserved		Access trap address bits A[15:10]: These bits along with SYSCFG D6h[1:0] and D7h[7:0] provide the 16-bit address of the port access that caused the SMI trap. D6h[2] indicates whether an I/O read or an I/O write access was trapped. D6h[3] gives the status of the SBHE# signal for the I/O instruction that was trapped.					

Table 4-98 PMU Registers Associated with DMA Trap

7	6	5	4	3	2	1	0
PCIDV1 58h			**DRQ Remap Base Address Register - Byte 0: Address Bits [7:0]**				**Default = 00h**
DRQ remap base address bits [7:0]: - The distributed DMA protocol requires DMA controller registers for each DMA channel to be individually mapped into I/O space outside the range claimed by ISA devices. Bits A[31:0] of this register specify that base; bits 6:0 are reserved (write 0) because the base address can fall only on 128 byte boundaries. The 82C700 logic uses this base address two ways: 1) to claim accesses to a PCMCIA DMA controller channel; 2) to forward accesses across the bridge to remote devices specified in the DMA Channel Selector Register.							
PCIDV1 59h			**DRQ Remap Base Address Register - Byte 1: Address Bits [15:8]**				**Default = 00h**
PCIDV1 5Ah			**DRQ Remap Base Address Register - Byte 2: Address Bits [23:16]**				**Default = 00h**
PCIDV1 5Bh			**DRQ Remap Base Address Register - Byte 3: Address Bits [31:24]**				**Default = 00h**

4.13.3.2 DMA Channel Selector Register

The register shown in Table 4-99 (PCIDV1 5Ch) is readable and writable, but performs no other function in the present silicon. PCI enumerator software selects "local" and "remote" ("on PCI") locations for each DMA channel through this register; a similar register exists on the 82C824. SMM code can then read this value to determine whether each channel is local or remote and can remap the access, or restart the cycle, accordingly.

4.13.3.3 System Configuration Registers

The PMU registers in Table 4-100 allow SMM code to initially enable the trap, and to later identify the source of the SMI.

Table 4-99 DMA Channel Selector Register

7	6	5	4	3	2	1	0
PCIDV1 5Ch			**DMA Channel Selector Register**				**Default = 00h**
Ch 7 (DMAC2): 0 = Local 1 = On PCI	Ch 6 (DMAC2): 0 = Local 1 = On PCI	Ch 5 (DMAC2): 0 = Local 1 = On PCI	Hardware Distributed DMA: 0 = Disable 1 = Enable	Ch 3 (DMAC1): 0 = Local 1 = On PCI	Ch 2 (DMAC1): 0 = Local 1 = On PCI	Ch 1 (DMAC1): 0 = Local 1 = On PCI	Ch 0 (DMAC1): 0 = Local 1 = On PCI

Table 4-100 PMU Register Bits Associated with DMA Trap

7	6	5	4	3	2	1	0
SYSCFG DDh			**PMU SMI Source Register 4 (Write 1 to Clear)**				**Default = 00h**
		PMI#37, DMA_ ACCESS: 0 = Inactive 1 = Active					
SYSCFG F5h			**PMU Event Register 8**				**Default = 00h**
						DMA_ACCESS PMI#37 SMI: 00 = Disable 11 = Enable	

4.14 IRQ Driveback Overview

The OPTi PCI IRQ Driveback cycle provides a clean and simple way to convey interrupt and DMA status information to the host. The protocol is reliable and does not in any way compromise PCI compatibility.

1. Whenever a PCI peripheral device must signal an IRQ or SMI# to the system, it asserts its REQ# line to the host for one PCI clock, deasserts it for one PCI clock, then asserts it again and keeps it low until acknowledged.
2. The host recognizes this sequence as a high-priority request and immediately removes all other bus grants (GNT# lines). Once the previous bus owner is off the bus, the host acknowledges the high-priority request with GNT# as usual.
3. The peripheral device logic runs an I/O write cycle to the IRQ Driveback address specified in the PCI configuration registers, and releases REQ#.
4. The host latches the information on AD[31:0] and sets the IRQ lines appropriately.
5. An optional second burst data cycle can take place to convey additional interrupt information.

PCI-type devices on the secondary side of bridge chips can use this same protocol to convey their interrupt requests through the bridge to the host. The format of the driveback cycle request is illustrated in the Figure 4-37. A second data phase is also possible.

4.14.1 Driveback Cycle Format

Table 4-101 and Table 4-102 illustrate the interrupt information indicated IRQ bits indicate whether that IRQ line is being driven high or low. The EN# bits indicate whether that IRQ is enabled to be changed or not. When the EN# bit is low, the value on the IRQ bit is valid. The device containing the central interrupt controller claims this I/O write cycle, and can then change its internal IRQ line state to match the value sent.

When a PCI device needs to generate an interrupt to the system, it runs a driveback cycle with the Enable bit low for each IRQ line under its control. For example, a device on PCI could run a driveback cycle with IRQ3 high and EN3# low to generate IRQ3 to the system. When the interrupt has been serviced and the device deasserts its interrupt, it starts another driveback cycle with IRQ3 low and EN3# low.

During both of these instances, if the device controls interrupts other than IRQ3, it must set its EN# bits low for **all** channels it controls, not just for the interrupt whose state has changed. The other IRQs must be driven with their previously used values.

Figure 4-37 IRQ Driveback Cycle High Priority Request

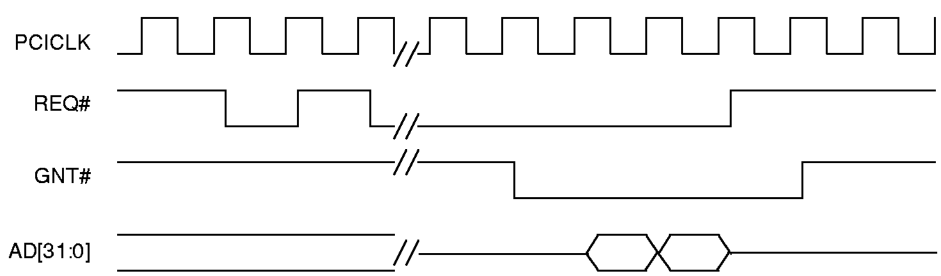


Table 4-101 Information Provided on a Driveback Cycle

Low Word	AD15	AD14	AD13	AD12	AD11	AD10	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
	IRQ15	IRQ14	IRQ13	IRQ12	IRQ11	IRQ10	IRQ9	IRQ8	IRQ7	IRQ6	IRQ5	IRQ4	IRQ3	IRQ2	IRQ1	IRQ0
High Word	AD31	AD30	AD29	AD28	AD27	AD26	AD25	AD24	AD23	AD22	AD21	AD20	AD19	AD18	AD17	AD16
	EN15#	EN14#	EN13#	EN12#	EN11#	EN10#	EN9#	EN8#	EN7#	EN6#	EN5#	EN4#	EN3#	EN2#	EN1#	EN0#

There is a convention for assignment of otherwise unusable IRQs:

- IRQ2 generates an SMI#. Note that the sense of IRQ2 is still active high. In this way, devices that use IRQ driveback can generate SMI# simply by routing their normal interrupt to IRQ2 without needing to change the polarity of the interrupt generation logic.
- IRQ13 generates an NMI. This feature allows PCI-to-ISA bridges such as the 82C825 chip to return the CHCK# signal from the ISA bus across the PCI bus. The sense of IRQ13 is active high.

Table 4-102 illustrates the format of the optional second data phase of the IRQ driveback cycle. This phase is presently reserved for returning the PCI interrupts and ACPI events. If the device needs to send back level-model interrupts, it bursts the information on the PCI clock following data phase one. The IRQ driveback address automatically increments to (base +4) per PCI requirements. It is also allowable for devices to drive back only phase 2, by directly accessing the (base +4) address.

Table 4-102 Information Provided on a Optional Data Phase 2 of IRQ Driveback Cycle

Low Word	**AD15**	**AD14**	**AD13**	**AD12**	**AD11**	**AD10**	**AD9**	**AD8**	**AD7**	**AD6**	**AD5**	**AD4**	**AD3**	**AD2**	**AD1**	**AD0**
	Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	ACPI3	ACPI2	ACPI1	ACPI0	PCIRQ 3	PCIRQ 2	PCIRQ 1	PCIRQ 0
High Word	**AD31**	**AD30**	**AD29**	**AD28**	**AD27**	**AD26**	**AD25**	**AD24**	**AD23**	**AD22**	**AD21**	**AD20**	**AD19**	**AD18**	**AD17**	**AD16**
	Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	EN ACPI3#	EN ACPI2#	EN ACPI1#	EN ACPI0#	ENP3#	ENP2#	ENP1#	ENP0#

4.14.2 Edge vs Level Mode, IRQ Polarity

The IRQs driven back in data phase 1 are interpreted as edge-mode interrupts, as expected for AT compatibility. The AD[15:0] signals are interpreted as active when high (1); the Enable (EN#) signals AD[31:16] are active when low (0).

In optional data phase 2, the PCIRQ[3:0]# bits are interpreted as level-mode interrupts by the host hardware. As with data phase 1, the controls indicated by AD[15:0] are interpreted as active when **high**; the Enable (EN#) controls on AD[31:16] are active when **low**. Note that PCI signals INTA-D# are active low by definition.

4.14.3 Host Handling of IRQ Driveback Information

The host chipset must handle the IRQ driveback information differently depending on whether the selected interrupt is sharable or not. Generally the ISA IRQ lines need no special consideration.

However, the INTA-D# lines can be shared by multiple devices on the PCI bus. Thus, one device could perform an IRQ driveback to set the INTx# line active for its purposes, while another device could follow immediately by setting the same INTx# line inactive. Therefore, the host is required to implement a counter in this case, so that it considers the line inactive only after it has received the same number of active-going drivebacks as it has inactive-going drivebacks.

A three-bit counter can be considered sufficient to handle the situation, since this would allow up to seven devices to chain to the same interrupt. It is unlikely that system requirements would exceed this number given the latency penalty incurred.

4.14.4 IRQ Driveback Support

FireStar uses the registers shown in Table 4-103 to control and monitor IRQ driveback.

Table 4-103 IRQ Driveback Control and Monitor Registers

7	6	5	4	3	2	1	0
PCIDV1 5Eh			**IRQ Scheme Management Register**				**Default = 00h**
End-of-Interrupt Holdoff bits [1:0]: The value of these bits selects the number of retries that will be forced on the PCI bus every time an attempt is made to write I/O Port 020h or 0A0h, where OCW2 of the interrupt controller is set. Multiple retries ensure that a device trying to generate an IRQ driveback will succeed before an EOI command takes effect. This feature eliminates the possibility that an EOI could be registered before a change in IRQ status gets back to the central interrupt controller.		IRQ driveback data readback selection at PCIDV1 60h-63h: 0 = 1st data phase 1 = 2nd data phase	Reserved				
PCIDV1 54h		**IRQ Driveback Address Register - Byte 0: Address Bits [7:0]**					**Default = 00h**
IRQ driveback protocol address bits [7:0]: - When an external device logic, such as the 82C824 PC Card Controller or the 82C814 Docking Controller, must generate an interrupt from any source, it follows the IRQ Driveback Protocol and toggles the REQ# line to the 82C700. Once it has the bus, it writes the changed IRQ information to the 32-bit I/O address specified in this register. The 82C700 interrupt controller claims this cycle and latches the new IRQ values. - This register defaults to a value of 00h, which disables the IRQ driveback scheme.							
PCIDV1 55h		**IRQ Driveback Address Register - Byte 1: Address Bits [15:8]**					**Default = 00h**
PCIDV1 56h		**IRQ Driveback Address Register - Byte 2: Address Bits [23:16]**					**Default = 00h**
PCIDV1 57h		**IRQ Driveback Address Register - Byte 3: Address Bits [31:24]**					**Default = 00h**
PCIDV1 60h		**IRQ Driveback Data Register - Byte 0: Data Bits [7:0]**					**Default = 00h**
IRQ Driveback Data Bits [7:0]: - Whenever the 82C700 receives an IRQ driveback cycle, it latches the entire 32-bit data value in this register. If any of the IRQs set active in this driveback are also programmed to generate an SMI (through the standard PMU register settings), SMM code can read this register to determine the exact driveback value written.							
PCIDV1 61h		**IRQ Driveback Data Register - Byte 1: Data Bits [15:8]**					**Default = 00h**
PCIDV1 62h		**IRQ Driveback Data Register - Byte 2: Data Bits [23:16]**					**Default = 00h**
PCIDV1 63h		**IRQ Driveback Data Register - Byte 3: Data Bits [31:24]**					**Default = 00h**

4.14.4.1 IRQ Scheme Management Register

SYSCFG F5h[3:2] allow the IRQs generated by the IRQ driveback scheme to either pass through unaffected or to be latched in PCIDV1 60h (IRQ Driveback Data Register) and an SMI generated. This feature provides a useful diagnostic tool but is not intended for general power management. The individual IRQ SMI enable bits provided in the PMU should be used for this function. Other bits associated with IRQ driveback are shown in Table 4-104.

Table 4-104 PCI IRQ Driveback Trap Bits

7	6	5	4	3	2	1	0
SYSCFG DDh			**PMU SMI Source Register 4 (Write 1 to Clear)**				**Default = 00h**
	PMI#38, CISA/PCI IRQ driveback trap: 0 = Inactive 1 = Active						
SYSCFG F5h			**PMU Event Register 8**				**Default = 00h**
				PCI IRQ driveback trap PMI#38 SMI: 00 = Disable 11 = Enable			

4.15 Power Management

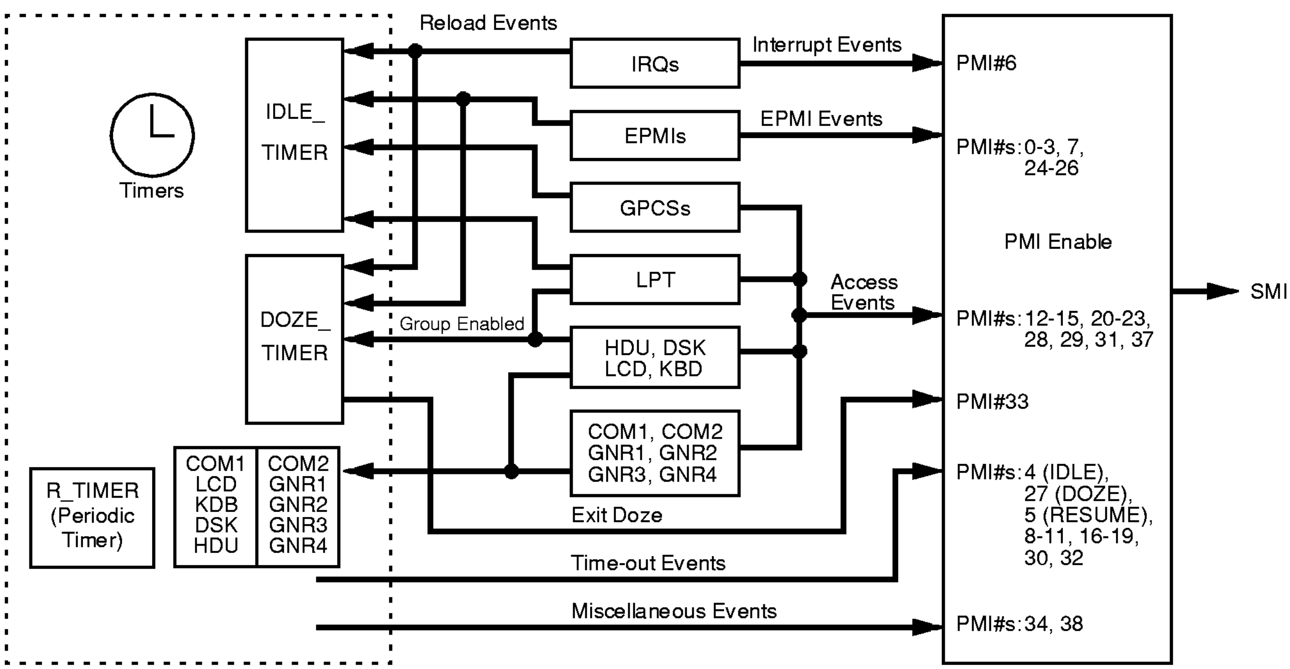
The synergistic incorporation of power management and system control features with the standard AT-subsystem controller of FireStar results in a compact design that handles multiple tasks with a simple, common interface. The power management unit (PMU) of FireStar is based on the implementation in the Viper-N+ Chipset and is designed to be register compatible for easy upgrading. The following subsections detail the operation of the PMU.

- The power management interrupt (PMI) scheme provides system management code with a quick means of identifying and handling events that affect power control and consumption.
- The PMU recognizes 33 separate PMI events. Within these events, many sub-events are also identifiable for a high degree of power management monitoring intelligence.
- Thirteen of the PMI events have individual timers to indicate inactivity time-out situations.
- Eight external inputs are available for monitoring asynchronous system events. These are in addition to the ISA IRQ lines that can also be monitored as power management events.
- PMI generation on access allows SMI code to intercept status queries to powered down devices that do not actually need to be restarted simply to return an “idle” status.
- An activity tracking register of ten events allows SMI or non-SMI applications a means of determining whether activity has occurred since the last time the register was checked. Polling of I/O activity can then be used instead of multiple SMIs for less significant events.
- Memory watchdog monitoring allows accesses to memory ranges (specified as programmed) to cause an SMI. ISA bus memory devices that are not being accessed can be programmed to cause a time-out SMI so that unused peripherals can be powered down.
- The PMU supports system-level low power Suspend, low power Suspend with zero volt CPU Suspend, or total system zero volt Suspend.
- Sixteen peripheral power control pins provide exceptional flexibility in peripheral device control.
- Real-time clock (RTC) alarm or modem ring can wake-up the system from the low power Suspend mode.
- Suspend current leakage control ensures that negligible power will be consumed in Suspend mode without additional external buffering.

4.15.1 Power Management Unit (PMU)

FireStar provides a large amount of programmable logic for managing system power control on the most precise of levels. The basic concepts of the FireStar power management scheme involve activity monitoring through time-outs and events. These concepts are illustrated in Figure 4-38 and described in detail in the following sections.

Figure 4-38 Activity Monitoring Block Diagram



4.15.1.1 Activity Monitoring

Activity monitoring is based on time-outs of countdown timers, the events that can be enabled to reload the timers and forestall a time-out, and the system management interrupts that can be generated in either case.

Timers

The thirteen FireStar timer registers all have _TIMER appended to their name.

The IDLE_TIMER times long periods of inactivity across all selected system peripherals to determine when a full power-down, called “Suspend” mode, is appropriate.

The DOZE_TIMER times short inactivity intervals (between keystrokes, for example) to put the system in an intermediate power-saving state called “Doze” mode.

The R_TIMER generates a periodic interrupt to allow system management code to poll for activity.

Ten other timers are available to monitor activity on specific peripheral devices so that system management software can shut each one down individually when possible while the rest of the system continues to operate.

Simply loading the timer with a countdown value presets the timers. Then the next access or interrupt event starts them counting down. A “dummy” access is needed in most cases to start the timer counting.

As each timer is clocked by its programmed source, it counts down to a time-out (zero) which generates a power management interrupt (PMI). The time-out PMI can, in turn, be enabled to generate an SMI (system management interrupt) on the SMI line that goes from FireStar to the CPU to trigger it into System Management Mode (SMM).

Events

Each timer has one or more events that can reload it with its original value, holding off the time-out. The events can be:

- Access Events
  - Those that are caused by CPU access to a certain I/O and or memory range associated with that timer.
- Internal Events
  - Triggered by ISA bus IRQ events or special external power management inputs (EPMIs).

All events can be enabled individually to generate a PMI; access events generate separately numbered PMIs, while interrupt events are combined into a single PMI (PMI#6).

As opposed to time-out caused PMIs, event PMIs can be enabled to:

- Reload the timer(s) and, if needed, restore the system clocks speed
- Generate an SMI
- Do both

Because of the flexibility of FireStar’s power management logic, the interaction among these mechanisms can become complex. It is important to bear in mind the basic goal of the logic in order to deal with it effectively.

Timer Clock Sources

FireStar’s logic implements thirteen distinct timer circuits. Each timer has a clock source associated with it. For all but the DOZE_TIMER, these are named SQW0, SQW1, SQW2, or SQW3; the DOZE_TIMER circuit works differently than the rest and is described separately in the Section 4.15.3, "Doze Mode". Table 4-106 shows the frequencies that can be applied to the rest of the _TIMER counters.

The SQW3 through SQW0 timings are based on the SQWIN input to FireStar’s logic, which is a 32KHz clock input to FireStar. SYSCFG 40h[6] (as shown in Table 4-105) provides a secondary range of time intervals and applies globally to all SQW3 through SQW0 selections.

Table 4-106 lists the range of time-out delays that can be achieved by selecting each SQWx + SYSCFG 40h[6] combination. The register bit locations for each timer are shown in Table 4-107. The timer source is selected by bit combinations:

- 00 = SQW0, 01 = SQW1, 10 = SQW2, 11 = SQW3

Table 4-105 Timer Control Bits

7	6	5	4	3	2	1	0
SYSCFG 40h			**PMU Control Register 1**				**Default = 00h**
	Global timer divide: 0 = ÷1 1 = ÷4						

Table 4-106 Time Interval Choices Applicable to _TIMER Settings

Choice	Bits	SYSCFG 40h[6] = 0: No Base Clock Divisor			SYSCFG 40h[6] = 1: Divide Base Clock by 4		
		Frequency	Decrement Timer Every:	Maximum Delay	Frequency	Decrement Timer Every:	Maximum Delay
SQW0	00	32768Hz	30.5µs	7.81ms	8.192KHz	0.122ms	31.25ms
SQW1	01	512Hz	1.95ms	0.5s	128Hz	7.8ms	2s
SQW2	10	16Hz	62.5ms	16s	4Hz	0.25s	64s
SQW3	11	0.5Hz	2s	8.5 min.	0.125Hz	8s	34 min.

Table 4-107 Timer Clock Source Selection Registers

7	6	5	4	3	2	1	0
SYSCFG 42h if AEh[7] = 0			**Clock Source Register 1**				**Default = 00h**
Clock source for GNR1_TIMER		Clock source for KBD_TIMER		Clock source for DSK_TIMER		Clock source for LCD_TIMER	
SYSCFG 42h if AEh[7] = 1			**Clock Source Register 1**				**Default = 00h**
Clock source for GNR5_TIMER		Reserved					
SYSCFG B2h if AEh[7] = 0			**Clock Source Register 2**				**Default = 00h**
Clock source for HDU_TIMER		Clock source for COM2_TIMER		Clock source for COM1_TIMER		Clock source for GNR2_TIMER	
SYSCFG B2h if AEh[7] = 1			**Clock Source Register 2**				**Default = 00h**
Reserved						Clock source for GNR6_TIMER	
SYSCFG 68h			**Clock Source Register 3**				**Default = 00h**
Clock source for R_TIMER		Clock source for IDLE_TIMER					
SYSCFG E6h if AEh[7] = 0			**Clock Source Register 4**				**Default = 70h**
				Clock source for GNR4_TIMER		Clock source for GNR3_TIMER	
SYSCFG E6h if AEh[7] = 1			**Clock Source Register 4**				**Default = 70h**
				Clock source for GNR8_TIMER		Clock source for GNR7_TIMER	
SYSCFG AEh			**GNR_ACCESS Feature Register**				**Default = 03h**
GNR set select: 0 = GNR1-4 1 = GNR5-8							

Time-Out Count and Time-Out SMI

The timer source registers listed in are used to load the initial time-out count. The following rules apply.

- A time-out count of five or greater indicates the countdown value. Time-out count values 1-4 should not be used (since the logic can take up to four clocks to reload a time-out count value, an invalid time-out could occur in the meantime).
- Writing a time-out count of 0 disables the timer.
- A dummy access in the appropriate address range for that timer triggers counting. From then on, additional accesses will reload the timer with its initial value and forestall a time-out.
- Reading the timer value will return only the value initially written, not the current count (except for R_TIMER, which **does** return the current count).

When a time-out occurs, it can do only one thing: trigger an SMI. Registers listed in Section 4.17.3, "Enabling of Events to Generate SMI", enable each time-out event individually to cause an SMI.

Note that the DOZE_TIMER Registers SYSCFG 41h and 79h contain the time count bits for the DOZE_TIMER and monitor selected IRQs and EPMIs. Unlike the other timer registers, the DOZE_TIMER uses its own time base selected through SYSCFG 41h[7:5]. A time-out generates PMI#27.

Table 4-108 Timer Registers

7	6	5	4	3	2	1	0
SYSCFG 44h			**LCD_TIMER Register**				**Default = 00h**
Time count byte for LCD_TIMER: Monitors LCD_ACCESS. Time-out generates PMI#8.							
SYSCFG 45h			**DSK_TIMER Register**				**Default = 00h**
Time count byte for DSK_TIMER: Monitors DSK_ACCESS. Time-out generates PMI#9.							
SYSCFG 46h			**KBD_TIMER Register**				**Default = 00h**
Time count byte for KBD_TIMER: Monitors KBD_ACCESS. Time-out generates PMI#10.							
SYSCFG 4Fh			**IDLE_TIMER Register**				**Default = 00h**
Time count byte for IDLE_TIMER: Monitors selected IRQs and EPMIs. Time-out generates PMI#4.							
SYSCFG 69h			**R_TIMER Register**				**Default = 00h**
- Time count byte for R_TIMER - starts to count after a non-zero write to this register. - Unlike the other timer registers, a read from this register returns the **current** count. - Time-out generates PMI#5.							
SYSCFG B4h			**HDU_TIMER Register**				**Default = 00h**
Time count byte for HDU_TIMER: Monitors HDU_ACCESS. Time-out generates PMI#19.							
SYSCFG B5h			**COM1_TIMER Register**				**Default = 00h**
Time count byte for COM1_TIMER: Monitors COM1_ACCESS. Time-out generates PMI#17.							
SYSCFG B6h			**COM2_TIMER Register**				**Default = 00h**
Time count byte for COM2_TIMER: Monitors COM2_ACCESS. Time-out generates PMI#18.							
SYSCFG 47h if AEh[7] = 0			**GNR1_TIMER Register**				**Default = 00h**
Time count byte for GNR1_TIMER: Monitors GNR1_ACCESS. Time-out generates PMI#11.							
SYSCFG 47h if AEh[7] = 1			**GNR5_TIMER Register**				**Default = 00h**
Time count byte for GNR5_TIMER: Monitors GNR5_ACCESS. Time-out generates PMI#11.							

Table 4-108 Timer Registers (cont.)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">SYSCFG B7h if AEh[7] = 0 — GNR2_TIMER Register — Default = 00h</td></tr>
<tr><td colspan="8">Time count byte for GNR2_TIMER: Monitors GNR2_ACCESS. Time-out generates PMI#16.</td></tr>
<tr><td colspan="8">SYSCFG B7h if AEh[7] = 1 — GNR6_TIMER Register — Default = 00h</td></tr>
<tr><td colspan="8">Time count byte for GNR6_TIMER: Monitors GNR6_ACCESS. Time-out generates PMI#16.</td></tr>
<tr><td colspan="8">SYSCFG E7h if AEh[7] = 0 — GNR3_TIMER Register — Default = 00h</td></tr>
<tr><td colspan="8">Time count byte for GNR3_TIMER: Monitors GNR3_ACCESS. Time-out generates PMI#29.</td></tr>
<tr><td colspan="8">SYSCFG E7h if AEh[7] = 1 — GNR7_TIMER Register — Default = 00h</td></tr>
<tr><td colspan="8">Time count byte for GNR7_TIMER: Monitors GNR7_ACCESS. Time-out generates PMI#29.</td></tr>
<tr><td colspan="8">SYSCFG E8h if AEh[7] = 0 — GNR4_TIMER Register — Default = 00h</td></tr>
<tr><td colspan="8">Time count byte for GNR4_TIMER: Monitors GNR4_ACCESS. Time-out generates PMI#30.</td></tr>
<tr><td colspan="8">SYSCFG E8h if AEh[7] = 1 — GNR8_TIMER Register — Default = 00h</td></tr>
<tr><td colspan="8">Time count byte for GNR8_TIMER: Monitors GNR8_ACCESS. Time-out generates PMI#30.</td></tr>
<tr><td colspan="8">SYSCFG 41h — DOZE_TIMER Register 2 — Default = 00h</td></tr>
<tr><td colspan="3">DOZE_0 time-out select:
000 = 2ms
001 = 4ms
010 = 8ms
011 = 32ms
100 = 128ms
101 = 512ms
110 = 2s
111 = 8s
Time-out generates PMI#27.</td><td colspan="3">Doze mode STPCLK# modulation (STPCLK# modulated by BCLK defined in SYSCFG E6h[7:6]):
000 = No Modulation (STPCLK# = 1)
001 = STPCLK# t_{hi} = 0.75 * 16 BCLKs
010 = STPCLK# t_{hi} = 0.5 * 16 BCLKs
011 = STPCLK# t_{hi} = 0.25 * 16 BCLKs
100 = STPCLK# t_{hi} = 0.125 * 16 BCLKs
101 = STPCLK# t_{hi} = 0.0625 * 16 BCLKs
110 = STPCLK# t_{hi} = 0.03125 * 32 BCLKs
111 = STPCLK# t_{hi} = 0.015625 * 64 BCLKs</td><td>ACCESS events reset Doze mode:
0 = Disable
1 = Enable</td><td>Doze control select:
0 = Hardware
1 = Software</td></tr>
<tr><td colspan="8">SYSCFG 79h — PMU Control Register 11 — Default = 00h</td></tr>
<tr><td colspan="3">DOZE_1 time-out select:
000 = No delay (Default)
001 = 1ms
010 = 4ms
011 = 16ms
100 = 64ms
101 = 256ms
110 = 1s
111 = 4s</td><td>PMI# event triggers exit from Doze mode if the PMI event is enabled to generate SMI:(1)
0 = No
1 = Yes</td><td>Reserved</td><td>PREQ# wake up Suspend:
0 = Disable
1 = Enable</td><td>CLKRUN# wake up Suspend:
0 = Disable
1 = Enable</td><td>ATCLK during Suspend:
0 = Run
1 = Stopped (overrides SYSCFG 66h[6])</td></tr>
<tr><td colspan="8">(1) For example, to let PMI#11 reset the Doze mode without generating SMI to the CPU, SYSCFG 5Ah[7:6] must = 11 and SYSCFG 5Bh[6] must = 1.</td></tr>
</table>

ACCESS Events

CPU memory and I/O instructions to peripheral devices cause power management events known as ACCESS events.

- Most ACCESS events generate an access PMI directly, which in turn can be enabled to activate the SMI input to the CPU so that the event can be serviced.
- These same events can be programmed to reload an associated countdown timer, thus preventing a time-out PMI from occurring.
- Still other ACCESS events can only cause a timer reload, and cannot directly generate an SMI.

Table 4-109 lists all of the ACCESS events, how the ACCESS can reload its associated timer and reload the IDLE_TIMER.

Note that enabled ACCESS events, except for the GNR and GPCS events, can be globally enabled to reload the DOZE_TIMER by setting SYSCFG 41h[1] = 1. Refer to Section 4.15.3, "Doze Mode", for details.

Serial (COMx) and Parallel Port (LPT) Access

Accesses to the LPT1, LPT2, and LPT3 I/O range group can be programmed to reload the IDLE_TIMER. For a greater degree of control, COM1 and COM2 can individually be enabled to cause COM1_ACCESS or COM2_ACCESS, reload the COM1_TIMER or COM2_TIMER, and reload the IDLE_TIMER.

Table 4-109 ACCESS Events and their Enabling Bit Locations

ACCESS Mnemonic	Monitored Range	ACCESS PMI#	Enable SMI on Current Access	Enable SMI on Next Access	Enable Reload of IDLE_ TIMER
LPT	Reads/writes in I/O address ranges 378-Fh, 278-Fh, and 3B8-Fh (LPT1, 2, and 3)	--	--	--	4Eh[5]
COM1	Reads/writes in I/O address range 3F8-Fh.	21	DEh[5]	DBh[1]	BEh[4]
COM2	Reads/writes in I/O address range 2F8-Fh.	22	DEh[6]	DBh[2]	BEh[5]
DSK	FDD accesses to I/O Port 3F5h and/or HDD accesses to 1F0-1F7h+3F6h. Bits 57h[5:4] determine which ranges apply.	13	DEh[1]	5Bh[1]	4Eh[1]
KBD	Reads/writes to I/O Ports 060h and 064h.	14	DEh[2]	5Bh[2]	4Eh[2]
LCD	Reads/writes in memory address range A0000-BFFFFh and/or I/O address range 3B0-3DFh. Bits 43h[7:6] and 5Fh[7:6]. determine which ranges apply.	12	DEh[0]	5Bh[0]	4Eh[0]
HDU	HDU accesses in the integrated IDE controller range: 1F0-7h + 3F6h (primary) or 170-7h + 376h (secondary). Bit ACh[2] determines which addresses apply.	23	DEh[7]	DBh[3]	BEh[2]
GPCS0	Defined in 4Ah[7:0], 4Bh[7:0], BFh[4,0]	--	--	--	4Eh[6]
GPCS1	Defined in 4Ch[7:0], 4Dh[7:0], BFh[5,1]	--	--	--	4Eh[7]
GPCS2	Defined in BCh[7:0], BDh[7:0], BFh[6,2]	--	--	--	BEh[6]
GPCS3	Defined in BAh[7:0], BBh[7:0], BFh[7,3]	--	--	--	BEh[7]
GNR1	Defined in bits 70h[7:0], 71h[7:0], 72h[7:0], 48h[7:0], 49h[7:0], and AEh[4,2,0]	15	DEh[3]	5Bh[3]	4Eh[3]
GNR2	Defined in bits 73h[7:0], 74h[7:0], 75h[7:0], B8h[7:0], B9h[7:0], and AEh[5,3,1]	20	DEh[4]	DBh[0]	BEh[3]
GNR3	Defined in bits E1h[7:0], E2h[7:0], and E5h[4,2,0]	31	E9h[1]	E9h[0]	4Eh[4]
GNR4	Defined in bits E3h[7:0], E4h[7:0], and E5h[5,3,1]	32	E9h[3]	E9h[2]	BEh[1]
DMA	Read/writes to I/O ports 000-00Fh, 400-40Fh, 0C0-0DFh, 4C0-4DFh, 4E0-4FFh, 080-08Fh, 480-48Fh.	37	F5h[1:0]	--	--

ISA Bus Floppy and Hard Drive Access

DSK_ACCESS can come from either or both of two separate access types. If enabled, the DSK_ACCESS reloads the DSK_TIMER and the IDLE_TIMER as well, if desired.

- Floppy accesses to generate DSK_ACCESS if SYSCFG 57h[5] = 0. The range of addresses that are to be monitored are determined by SYSCFG D6h[7].
- Hard disk accesses to 1F0-1F7h and 3F6h generate DSK_ACCESS if SYSCFG 57h[4] = 0. Both ISA bus IDE accesses and PCI bus IDE accesses will generate the access event.

Two separate and independent hard disk drives can be managed if the primary drive is on the ISA bus or PCI bus and the secondary drive is managed by the integrated IDE controller. Refer to the HDU_ACCESS event regarding access events from the integrated local bus IDE controller.

Integrated Controller Hard Drive Access

Accesses to the integrated hard disk controller, in the primary range 1F0-1F7h and 3F6-3F7h or the secondary range 170-177h and 376-377h can cause HDU_ACCESS, reload the HDU_TIMER, and reload the IDLE_TIMER. SYSCFG FCh and FDh determine which addresses apply.

HDU_ACCESS is based solely on the decoding for the internal IDE controller. It is independent of the DSK_ACCESS decoding. Therefore, SYSCFG 57h[4] does not affect HDU_ACCESS. DSK_ACCESS can continue to monitor both floppy disk and primary external hard disk accesses if desired.

Selectable DSK_ACCESS Address Range

FireStar can power manage four IDE drives handled by the local bus IDE controller. Four independent timers are available, DSK_TIMER, HDU_TIMER, GNR2_TIMER, and GNR3_TIMER. Since IDE drives are accessed two per cable at the same I/O address range, the PMU logic uses the IDE drive's register bit 1F6h[4] to distinguish between drives 0 and 1, and 176h[4] to distinguish between drives 2 and 3. The last value written to this bit determines whether all other accesses in that range reload the first or second drive timer for that cable.

Note that the PMI events themselves must be enabled, as before, through the appropriate bits SYSCFG 5Ah[3:2] for DSK_ACCESS, and SYSCFG D8h[7:6] for HDU_ACCESS. Also, SYSCFG 57h[4] is independent of these new bits, and still monitors only ISA bus devices at the primary I/O range.

Keyboard Access

Keyboard accesses to I/O Ports 060h and 064h can cause KBD_ACCESS, reload the KBD_TIMER, and reload the IDLE_TIMER.

LCD Controller Access

Video controller accesses are to I/O Ports 3B0-3DFh and to memory locations A0000-BFFFFh if not masked by SYSCFG 43h[7:6].

The enabled accesses cause LCD_ACCESS, reload the LCD_TIMER, and reload the IDLE_TIMER if not masked in SYSCFG 5Fh[7:6].

Table 4-110 shows the PMU control registers discussed above.

Table 4-110 PMU Control Registers

7	6	5	4	3	2	1	0
SYSCFG 57h			**PMU Control Register 5**				**Default = 08h**
		DSK_ACCESS includes FDD: 0 = Yes 1 = No	DSK_ACCESS includes HDD: 0 = Yes 1 = No				
SYSCFG D6h			**PMU Control Register 10**				**Default = 00h**
DSK_ACCESS: 0 = 3F5h only 1 = All FDC Ports (3F2,4,5,7 & 372,4,5,7h)							

Table 4-110 PMU Control Registers (cont.)

7	6	5	4	3	2	1	0
SYSCFG FCh			**IDE Power Management Assignment Register 1**				**Default = 33h**
IDE Drive 1 I/O access reloads GNR4_TIMER: 0 = No 1 = Yes	IDE Drive 1 I/O access reloads GNR3_TIMER: 0 = No 1 = Yes	IDE Drive 1 I/O access reloads HDU_TIMER: 0 = No 1 = Yes	IDE Drive 1 I/O access reloads DSK_TIMER: 0 = No 1 = Yes	IDE Drive 0 I/O access reloads GNR4_TIMER: 0 = No 1 = Yes	IDE Drive 0 I/O access reloads GNR3_TIMER: 0 = No 1 = Yes	IDE Drive 0 I/O access reloads HDU_TIMER: 0 = No 1 = Yes	IDE Drive 0 I/O access reloads DSK_TIMER: 0 = No 1 = Yes
Note: If a bus mastering drive is used, DDRQ will also reload the enabled timer(s).							
SYSCFG FDh			**IDE Power Management Assignment Register 2**				**Default = 33h**
IDE Drive 3 I/O access reloads GNR4_TIMER: 0 = No 1 = Yes	IDE Drive 3 I/O access reloads GNR3_TIMER: 0 = No 1 = Yes	IDE Drive 3 I/O access reloads HDU_TIMER: 0 = No 1 = Yes	IDE Drive 3 I/O access reloads DSK_TIMER: 0 = No 1 = Yes	IDE Drive 2 I/O access reloads GNR4_TIMER: 0 = No 1 = Yes	IDE Drive 2 I/O access reloads GNR3_TIMER: 0 = No 1 = Yes	IDE Drive 2 I/O access reloads HDU_TIMER: 0 = No 1 = Yes	IDE Drive 2 I/O access reloads DSK_TIMER: 0 = No 1 = Yes
Note: If a bus mastering drive is used, DDRQ will also reload the enabled timer(s).							
SYSCFG 5Ah if AEh[7] = 0			**PMU Event Register 3**				**Default = 00h**
				DSK_TIMER PMI#9 DSK_ACCESS PMI#13: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI			
SYSCFG D8h if AEh[7] = 0			**PMU Event Register 5**				**Default = 00h**
HDU_TIMER PMI#19 HDU_ACCESS PMI#23: 00 = Disable 01 = Reserved 01 = Reserved 11 = SMI							
SYSCFG 43h			**PMU Control Register 3**				**Default = 00h**
LCD_ACCESS includes I/O range 3B0h-3DFh: 0 = Yes 1 = No	LCD_ACCESS includes memory A0000-BFFFFh: 0 = Yes 1 = No						
SYSCFG 5Fh			**PMU Control Register 6**				**Default = 00h**
LCD_ACCESS includes ISA bus video access: 0 = Yes 1 = No	LCD_ACCESS includes local (PCI) bus video access: 0 = No 1 = Yes						

Chip Select Generation (GPCS) Access

The GPCS[3:0]# lines can be programmed to generate a chip select based on either memory or I/O decoding of reads and/or writes. Even if the external logic necessary to implement the chip select lines is not in place, the chip select events themselves can be individually enabled to reload the IDLE_TIMER through SYSCFG 4Eh[7:6] and BEh[7:6] (shown in Table 4-111).

Table 4-111 GPCS Access Register Bits

7	6	5	4	3	2	1	0
SYSCFG 4Eh if AEh[7] = 0			**Idle Reload Event Enable Register 1**				**Default = 00h**
GPCS1#_ ACCESS: 0 = Disable 1 = Enable	GPCS0#_ ACCESS: 0 = Disable 1 = Enable						
SYSCFG BEh if AEh[7] = 0			**Idle Reload Event Enable Register 2**				**Default = 00h**
GPCS3#_ ACCESS: 0 = Disable 1 = Enable	GPCS2#_ ACCESS: 0 = Disable 1 = Enable						

General Purpose (GNR) Access

Four programmable ranges, GNR1, GNR2, GNR3, and GNR4, are provided, each with its own separate timer, to allow any four I/O or memory ranges to be monitored (refer to Table 4-112). As an example, the COM3 I/O range 3E8-3EFh could be monitored for reads and writes in order to determine whether the connected UART was in active use. As another example, a network card that uses memory in the D800-DFFFh half-segment could be monitored to determine whether the memory is being accessed regularly and, if not, a query could be sent through the network to ensure that the connection was still valid.

Memory Watchdog Feature

FireStar's general purpose access register sets, GNR1, GNR2, GNR3, and GNR4, can be monitored for activity and can generate an SMI when no activity has occurred in a given amount of time. As an option, either or both of these register sets can be assigned to monitor memory space instead. In this case, instead of the bit values corresponding to I/O address bits A[9:0], the values correspond to memory address bits A[23:14]. The bits that select I/O read or I/O write cycles instead indicate memory read or memory write cycles.

Example:
To monitor memory write activity in the 16KB block from CC00:0 to CC00:3FFF requires first viewing the CC00 segment value as:

- 0000 1100 1100 0000 0000 0000

to determine the value of the upper ten bits, CA[23:14], which is:

- 0000110011

to write into the A[9:0] GNR address decode bits. The bits are set by writing:

- SYSCFG E1h (A[8:1]) = 00011001b, or 19h
- SYSCFG E2h (A9 + write decode + read decode + A[5:1] mask bits) = 01000000b, or 40h
- SYSCFG E5h (GNR4 cycle + GNR3 cycle + GNR4 A0 + GNR3 A0 + GNR4 A0 mask + GNR3 A0 mask) = 011000b, or 18h (GNR4 values must also be considered).

The timer values must then be entered, the PMI enabled, and then a dummy write access must be made to the CC000-CFFFFh range to start GNR3_TIMER. If no accesses are occurring, the timer will eventually expire and generate an SMI. If enabled, the next write access to this range will also cause an SMI and will reload the timer.

Of the four general purpose access register sets, two sets, GNR1 and GNR2, provide granularity for the memory watchdog function to monitor a minimum range of four bytes, while GNR3 and GNR4 provide granularity to monitor accesses in a 64KB range. If SYSCFG A0h[7] = 1 such that upper address bits must be zero, the GNR1 and GNR2 registers still decode the full 16 bits of the address as long as those upper bits are not masked off (default).

Extra General Purpose Decode Ranges

FireStar provides four additional general purpose ranges, GNR5-8, for monitoring of system peripheral devices. These ranges are accessible through the existing register set in place of GNR1-4, but internally they are distinct and separate from GNR1-4. Both sets of ranges can be used simultaneously for a total of eight programmable ranges.

To initially program the registers, software sets SYSCFG AEh[7] = 1. From this point on, all GNR1 register bits access GNR5 bits instead; all GNR2 register bits access GNR6 instead; and so on. Once programming is complete, software should clear SYSCFG AEh[7].

When access in a GNR5-8 range is decoded, the chipset will generate an SMI. Software proceeds as usual to read SYSCFG 5Ch, 5Dh, DCh, DDh, and EAh to determine the source of the SMI. Since SYSCFG AEh[7] = 0 at this point, SMI status for GNR1-4 will be returned in the appropriate bits of SYSCFG 5Dh, DCh, and EAh.

Software then sets SYSCFG AEh[7] =1, and reads SYSCFG 5Dh, DCh, and EAh again. This time, the bits will indicate SMI activity in GNR5-8 instead of GNR1-4. Writing '1' back to the SMI source will clear only GNR5-8 as long as SYSCFG AEh[7] = 1.

DMA Controller Access
Access trapping of the DMA controller registers is described in Section 4.13.3, "Software Distributed DMA Support".

Table 4-112 General Purpose Access Registers

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">SYSCFG 70h</td><td colspan="4">GNR1 Base Address Register 1</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="8">GNR1_ACCESS base address: A[13:6] for memory watchdog or A[15:10] for I/O (right-aligned).</td></tr>
<tr><td colspan="2">SYSCFG 71h</td><td colspan="4">GNR1 Control Register 1</td><td colspan="2">Default = FFh</td></tr>
<tr><td colspan="8">GNR1_ACCESS mask bits: Mask for A[13:6] for memory watchdog or mask for A[15:10] for I/O (right-aligned).</td></tr>
<tr><td colspan="2">SYSCFG 72h</td><td colspan="4">GNR1 Control Register 2</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="4">GNR1_ACCESS base address:
A[5:2] for memory watchdog or ignored for I/O.</td><td colspan="4">GNR1_ACCESS mask bits:
Mask for A[5:2] for memory watchdog or mask for A[9:6] for I/O.</td></tr>
<tr><td colspan="2">SYSCFG 48h if AEh[7] = 0</td><td colspan="4">GNR1 Base Address Register</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="8">GNR1_ACCESS base address: A[8:1] (I/O) or A[22:15] (Memory)</td></tr>
<tr><td colspan="2">SYSCFG 48h if AEh[7] = 1</td><td colspan="4">GNR5_Timer Base Address Register</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="8">GNR5_TIMER base address: A[8:1] (I/O)</td></tr>
<tr><td colspan="2">SYSCFG 49h if AEh[7] = 0</td><td colspan="4">GNR1 Control Register</td><td colspan="2">Default = 00h</td></tr>
<tr><td>GNR1 base address:
A9 (I/O)
A23 (Memory)</td><td>Write decode:
0 = Disable
1 = Enable</td><td>Read decode:
0 = Disable
1 = Enable</td><td colspan="5">GNR1 mask bits for address A[5:1] (I/O) or A[19:15] memory:
A 1 in a particular bit means that the corresponding bit at SYSCFG 48h[4:0] is not compared. This is used to determine address block size.</td></tr>
<tr><td colspan="2">SYSCFG 49h if AEh[7] = 1</td><td colspan="4">GNR5_Timer Control Register</td><td colspan="2">Default = 00h</td></tr>
<tr><td>Base address:
A9 (I/O)</td><td>Write decode:
0 = Disable
1 = Enable</td><td>Read decode:
0 = Disable
1 = Enable</td><td colspan="5">GNR5 base address A[5:1] (I/O)</td></tr>
<tr><td colspan="2">SYSCFG 73h</td><td colspan="4">GNR2 Base Address Register 1</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="8">GNR2_ACCESS base address: A[13:6] for memory watchdog or A[15:10] for I/O (right-aligned).</td></tr>
<tr><td colspan="2">SYSCFG 74h</td><td colspan="4">GNR2 Control Register 1</td><td colspan="2">Default = FFh</td></tr>
<tr><td colspan="8">GNR2_ACCESS mask bits: Mask for A[13:6] for memory watchdog or mask for A[15:10] for I/O (right-aligned).</td></tr>
<tr><td colspan="2">SYSCFG 75h</td><td colspan="4">GNR2 Control Register 2</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="4">GNR2_ACCESS base address:
A[5:2] for memory watchdog or ignored for I/O.</td><td colspan="4">GNR2_ACCESS mask bits:
Mask for A[5:2] for memory watchdog or mask for A[9:6] for I/O.</td></tr>
</table>

Table 4-112 General Purpose Access Registers (cont.)

7	6	5	4	3	2	1	0
SYSCFG B8h if AEh[7] = 0			**GNR2 Base Address Register**				**Default = 00h**
GNR2_ACCESS base address: A[8:1] (I/O) or A[22:15] (Memory)							
SYSCFG B8h if AEh[7] = 1			**GNR6 Base Address Register**				**Default = 00h**
GNR6_Timer base address: A[8:1] (I/O)							
SYSCFG B9h if AEh[7] = 0			**GNR2 Control Register**				**Default = 00h**
GNR2 base address: A9 (I/O) A23 (Memory)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	GNR2 mask bits for address A[5:1] (I/O) or A[19:15] memory: A 1 in a particular bit means that the corresponding bit at B8h[4:0] is not compared. This is used to determine address block size.				
SYSCFG B9h if AEh[7] = 1			**GNR6 Control Register**				**Default = 00h**
GNR6 base address: A9 (I/O)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	GNR6 mask bits for address A[5:1] (I/O)				
SYSCFG AEh			**GNR_ACCESS Feature Register**				**Default = 03h**
GNR set select: 0 = GNR1-4 1 = GNR5-8	Reserved	GNR2 cycle decode type: 0 = I/O 1 = Memory	GNR1 cycle decode type: 0 = I/O 1 = Memory	GNR2 base address: A0 (I/O) A14 (Memory)	GNR1 base address: A0 (I/O) A14 (Memory)	GNR2 mask bit: A0 (I/O) A14 (Memory)	GNR1 mask bit: A0 (I/O) A14 (Memory)
SYSCFG 7Ah			**GNR3 Base Address Register 1**				**Default = 00h**
GNR3_ACCESS base address: A[13:6] for memory watchdog or A[15:10] for I/O (right-aligned).							
SYSCFG 7Bh			**GNR3 Control Register 1**				**Default = FFh**
GNR3_ACCESS mask bits: Mask for A[13:6] for memory watchdog or mask for A[15:10] for I/O (right-aligned).							
SYSCFG 7Ch			**GNR3 Control Register 2**				**Default = 00h**
GNR3_ACCESS base address: A[5:2] for memory watchdog or ignored for I/O.				GNR3_ACCESS mask bits: Mask for A[5:2] for memory watchdog or mask for A[9:6] for I/O.			
SYSCFG E1h if AEh[7] = 0			**GNR3 Base Address Register**				**Default = 00h**
GNR3_ACCESS base address: A[8:1] (I/O) or A[22:15] (Memory)							
SYSCFG E1h if AEh[7] = 1			**GNR7 Base Address Register**				**Default = 00h**
GNR7_ACCESS base address: A[8:1] (I/O)							
SYSCFG E2h if AEh[7] = 0			**GNR3 Control Register**				**Default = 00h**
GNR3 base address: A9 (I/O) A23 (Memory)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	GNR3 mask bits for address A[5:1] (I/O) or A[19:15] memory: A 1 in a particular bit means that the corresponding bit at SYSCFG E1h[4:0] is not compared. This is used to determine address block size.				
SYSCFG E2h if AEh[7] = 1			**GNR7 Control Register**				**Default = 00h**
GNR7 base address: A9 (I/O)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	GNR7 mask bits for address A[5:1] (I/O)				

Table 4-112 General Purpose Access Registers (cont.)

7	6	5	4	3	2	1	0
SYSCFG 7Dh			**GNR4 Base Address Register 1**				**Default = 00h**
GNR4_ACCESS base address: A[13:6] for memory watchdog or A[15:10] for I/O (right-aligned).							
SYSCFG 7Eh			**GNR4 Control Register 1**				**Default = FFh**
GNR4_ACCESS mask bits: Mask for A[13:6] for memory watchdog or mask for A[15:10] for I/O (right-aligned).							
SYSCFG 7Fh			**GNR4 Control Register 2**				**Default = 00h**
GNR4_ACCESS base address: A[5:2] for memory watchdog or ignored for I/O.				GNR4_ACCESS mask bits: Mask for A[5:2] for memory watchdog or mask for A[9:6] for I/O.			
SYSCFG E3h if AEh[7] = 0			**GNR4 Base Address Register**				**Default = 00h**
GNR4_ACCESS base address: A[8:1] (I/O) or A[22:15] (Memory)							
SYSCFG E3h if AEh[7] = 1			**GNR8 Base Address Register**				**Default = 00h**
GNR8_ACCESS base address: A[8:1] (I/O)							
SYSCFG E4h if AEh[7] = 0			**GNR4 Control Register**				**Default = 00h**
GNR4 base address: A9 (I/O) A23 (Memory)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	GNR4 mask bits for address A[5:1] (I/O) or A[19:15] memory: A 1 in a particular bit means that the corresponding bit at SYSCFG E3h[4:0] is not compared. This is used to determine address block size.				
SYSCFG E4h if AEh[7] = 1			**GNR8 Control Register**				**Default = 00h**
GNR8 base address: A9 (I/O)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	GNR8 mask bits for address A[5:1] (I/O)				
SYSCFG E5h			**GNR_ACCESS Feature Register 2**				**Default = 03h**
Reserved	Reserved	GNR4 cycle decode type: 0 = I/O 1 = Memory	GNR3 cycle decode Type: 0 = I/O 1 = Memory	GNR4 base address: A0 (I/O) A14 (Memory)	GNR3 base address: A0 (I/O) A14 (Memory)	GNR4 mask bit: A0 (I/O) A14 (Memory)	GNR3 mask bit: A0 (I/O) A14 (Memory)

4.15.1.2 Activity Tracking Registers

The activity tracking registers at SYSCFG DFh and E0h allow events to be flagged even if they are not programmed to generate an SMI. In this way, code can check whether a keystroke occurred since the last time the register was checked, for example, without actually generating an SMI for every keystroke.

The activity tracking registers record activity on all ten ACCESS events. No type of enabling is needed for any of these events to be registered. Reading this register returns flags indicating whether any of the events have taken place and automatically resets the entire register. The register can be written if desired to set the selected bits. In this way, a read-modify-write code sequence can be used to clear selected bits only.

Table 4-113 Activity Tracking Registers

7	6	5	4	3	2	1	0
SYSCFG DFh if AEh[7] = 0			**Activity Tracking Register**				**Default = 00h**
HDU_ ACCESS activity: 0 = No 1 = Yes	COM2_ ACCESS activity: 0 = No 1 = Yes	COM1_ ACCESS activity: 0 = No 1 = Yes	GNR2_ ACCESS activity: 0 = No 1 = Yes	GNR1_ ACCESS activity: 0 = No 1 = Yes	KBD_ ACCESS activity: 0 = No 1 = Yes	DSK_ ACCESS activity: 0 = No 1 = Yes	LCD_ ACCESS activity: 0 = No 1 = Yes
SYSCFG DFh if AEh[7] = 1			**Activity Tracking Register**				**Default = 00h**
Reserved			GNR6_ ACCESS activity: 0 = No 1 = Yes	GNR5_ ACCESS activity: 0 = No 1 = Yes	Reserved		
SYSCFG E0h if AEh[7] = 0			**Activity Tracking Register 1**				**Default = 00h**
Reserved						GNR4_ ACCESS activity: 0 = No 1 = Yes	GNR3_ ACCESS activity: 0 = No 1 = Yes
SYSCFG E0h if AEh[7] = 1			**Activity Tracking Register 1**				**Default = 00h**
Reserved						GNR8_ ACCESS activity: 0 = No 1 = Yes	GNR7_ ACCESS activity: 0 = No 1 = Yes

4.15.1.3 Reloading IDLE_TIMER

FireStar provides the IDLE_TIMER to monitor system-wide activity: I/O and memory accesses by the CPU, IRQs from ISA bus peripherals, and EPMIs from power control and management subsystems. The occurrence of an enabled event in any one of these areas will reload the IDLE_TIMER. Once there is inactivity for a sufficiently long time, the IDLE_TIMER will expire.

Expiration of the IDLE_TIMER generates PMI#4, which can be enabled to generate an SMI to inform system management code that the system is idle and that entry into the Suspend mode is appropriate. Refer to the Section 4.15.5.1, "Suspend Mode" for complete information. Expiration of the IDLE_TIMER cannot cause automatic (hardware-controlled) entry into the Suspend mode, since important CPU processing could be interrupted.

The register bits that enable each event individually to reload the IDLE_TIMER and forestall entry into the Suspend mode are shown in Table 4-114. SYSCFG 63h and A3h are write-only; reads return no useful information.

Table 4-114 Idle Reload Source Registers

7	6	5	4	3	2	1	0
SYSCFG 4Eh if AEh[7] = 0			**Idle Reload Event Enable Register 1**				**Default = 00h**
GPCS1#_ ACCESS: 0 = Disable 1 = Enable	GPCS0#_ ACCESS: 0 = Disable 1 = Enable	LPT_ ACCESS: 0 = Disable 1 = Enable	GNR3_ ACCESS: 0 = Disable 1 = Enable	GNR1_ ACCESS: 0 = Disable 1 = Enable	KBD_ ACCESS: 0 = Disable 1 = Enable	DSK_ ACCESS: 0 = Disable 1 = Enable	LCD_ ACCESS: 0 = Disable 1 = Enable
SYSCFG 4Eh if AEh[7] = 1			**Idle Reload Event Enable Register 1**				**Default = 00h**
Reserved			GNR7: 0 = Disable 1 = Enable	GNR5: 0 = Disable 1 = Enable	Reserved	Any PCI requests: 0 = Disable 1 = Enable	Reserved
SYSCFG BEh if AEh[7] = 0			**Idle Reload Event Enable Register 2**				**Default = 00h**
GPCS3#_ ACCESS: 0 = Disable 1 = Enable	GPCS2#_ ACCESS: 0 = Disable 1 = Enable	COM2_ ACCESS: 0 = Disable 1 = Enable	COM1_ ACCESS: 0 = Disable 1 = Enable	GNR2_ ACCESS: 0 = Disable 1 = Enable	HDU_ ACCESS: 0 = Disable 1 = Enable	GNR4_ ACCESS: 0 = Disable 1 = Enable	Override SYSCFG 68h[3:2]: 0 = No 1 = Recover time 1s
SYSCFG BEh if AEh[7] = 1			**Idle Reload Event Enable Register 2**				**Default = 00h**
Reserved				GNR6_ ACCESS: 0 = Disable 1 = Enable	Reserved	GNR8_ ACCESS: 0 = Disable 1 = Enable	Reserved
SYSCFG 63h			**Idle Time-Out Select Register 1**				**Default = 00h**
EPMI0# Level-trig'd: 0 = Disable 1 = Enable	IRQ13: 0 = Disable 1 = Enable	IRQ8: 0 = Disable 1 = Enable	IRQ7: 0 = Disable 1 = Enable	IRQ5: 0 = Disable 1 = Enable	IRQ4: 0 = Disable 1 = Enable	IRQ3: 0 = Disable 1 = Enable	IRQ0: 0 = Disable 1 = Enable
SYSCFG A3h			**Idle Time-Out Select Register 2**				**Default = 00h**
IRQ15: 0 = Disable 1 = Enable	IRQ14: 0 = Disable 1 = Enable	IRQ12: 0 = Disable 1 = Enable	IRQ11: 0 = Disable 1 = Enable	IRQ10: 0 = Disable 1 = Enable	IRQ9: 0 = Disable 1 = Enable	IRQ6: 0 = Disable 1 = Enable	IRQ1: 0 = Disable 1 = Enable
SYSCFG FBh			**DMA Idle Reload Register**				**Default = 00h**
DRQ7 reloads IDLE_TIMER: 0 = No 1 = Yes	DRQ6 reloads IDLE_TIMER: 0 = No 1 = Yes	DRQ5 reloads IDLE_TIMER: 0 = No 1 = Yes	IDE DDRQ reloads IDLE_TIMER:[1] 0 = No 1 = Yes	DRQ3 reloads IDLE_TIMER: 0 = No 1 = Yes	DRQ2 reloads IDLE_TIMER: 0 = No 1 = Yes	DRQ1 reloads IDLE_TIMER: 0 = No 1 = Yes	DRQ0 reloads IDLE_TIMER: 0 = No 1 = Yes

(1) Bit 4 controls whether the DDRQ line from bus mastering IDE drives can reload the timers. The bit controls DDRQ from both cables, so enabling the reload feature on any one bus mastering drive enables it for all present.

4.15.1.4 External PMI Events

FireStar's logic can monitor a variety of inputs that are directly related to low-power, battery-operated system designs. Table 4-115 lists the external power management input (EPMI) pins provided. Note that all pins included here are considered external PMI pins, not just pins EPMI[3:0]#.

EPMI Programming

The registers listed in Table 4-116 are used to initialize the EPMI pins and enable them to cause PMI events. The table also lists related EPMI programming registers.

Emergency Overtemp Sense Enable - Setting SYSCFG A1h[2] = 1 allows a level on the EPMI1# pin to force the chip into cool-down clocking mode as set by the thermal management registers. The thermal management feature itself does not need to be enabled to use this sense function. The polarity of the input is determined by SYSCFG 40h[2]. Once written to 1, this bit cannot be cleared without a hardware reset.

EPMI[1:0]# Status Latch - Setting SYSCFG A1h[0] = 1 allows the EPMI[1:0]# PMI events to be latched. The status returned by SYSCFG 5Ch[2:1] are **not** latched. Writing these same bits to 1 clears the status bits.

Table 4-115 External PMI Source Summary

Name	Description
LOWBAT	Activity on Low Battery pin
LLOWBAT	Activity on Very Low Battery pin
EPMI0#	Activity on External Power Management Input 0
EPMI1#	Activity on External Power Management Input 1
EPMI2#	Activity on External Power Management Input 2
EPMI3#	Activity on External Power Management Input 3
RESUME	SUS/RES input has been toggled while in Suspend
SUSPEND	SUS/RES input has been toggled while system is active
RINGI	Activity detected on RINGI

Table 4-116 EPMI Programming Registers

7	6	5	4	3	2	1	0
SYSCFG 40h			**PMU Control Register 1**				**Default = 00h**
		LLOWBAT polarity: 0 = Active high 1 = Active low	LOWBAT polarity: 0 = Active high 1 = Active low		EPMI1# polarity: 0 = Active high 1 = Active low	EPMI0# polarity: 0 = Active high 1 = Active low	
SYSCFG DBh if AEh[7] = 0			**Next Access Event Generation Register 2**				**Default = 00h**
		External EPMI3# pin polarity: 0 = Active high 1 = Active low	External EPMI2# pin polarity: 0 = Active high 1 = Active low				
SYSCFG 43h			**PMU Control Register 3**				**Default = 00h**
		LOWBAT pin sample rate: 00 = 32s 10 = 128s 01 = 64s 11 = Rsrvd A PMI is generated each time LOWBAT is sampled active.					

Table 4-116 EPMI Programming Registers (cont.)

7	6	5	4	3	2	1	0
SYSCFG 50h			**PMU Control Register 4**				**Default = 00h**
							Start Suspend (WO): 1 = Enter Suspend mode
SYSCFG 5Ch			**PMI SMI Source Register 1 (Write 1 to Clear)**				**Default = 00h**
				PMI#3, LOWBAT: 0 = Not Active 1 = Active	PMI#2, EPMI1#: 0 = Not Active 1 = Active	PMI#1, EPMI0#: 0 = Not Active 1 = Active	PMI#0, LLOWBAT: 0 = Not Active 1 = Active
SYSCFG 61h			**Debounce Register**				**Default = 00h**
LOWBAT, LLOWBAT debounce rate select: 00 = No debounce 01 = 250μs 10 = 8ms 11 = 500ms		SUS/RES# debounce rate select: 00 = Active low, edge-trig'd PMI 01 = Active low, level-controlled PMI 10 = Active high, level-sampled PMI in 16ms 11 = Active high, level-sampled PMI in 32ms (See Section 4.15.8.2, "SUS/RES and RINGI Events")					
SYSCFG A1h			**Feature Control Register 2**				**Default = 00h**
					Emerg. over-temp sense: 0 = Disable 1 = Enable		EPMI[1:0]# status latch: 0 = Dynamic 1 = Latched

Notes: 1) EPMI0# and EPMI1# need to be asserted until recognized by its SMI service routine, since these PMIs are not latched unless SYSCFG A1h[0] = 1.

2) If EPMI0# and EPMI1# are used to place the system into Suspend, the EPMIx signal must be negated before the Suspend command (setting bit SYSCFG 50h[0] = 1) is written.

4.15.1.5 Power Management Event Status

The power management input pins can be monitored for their instantaneous state in FireStar. This feature can be used to poll for power management status without generating an SMI.

The bits of the Power Management Event Status Register return instantaneous pin status; the state is not latched. Table 4-117 gives the bit definitions for SYSCFG DAh.

Table 4-117 Power Management Event Status

7	6	5	4	3	2	1	0
SYSCFG DAh			**Power Management Event Status Register (RO)**				**Default = 00h**
Reserved		LOWBAT state: 0 = Low 1 = High	LLOWBAT state: 0 = Low 1 = High	EPMI3# state: 0 = Low 1 = High	EPMI2# state: 0 = Low 1 = High	EPMI1# state: 0 = Low 1 = High	EPMI0# state: 0 = Low 1 = High

4.15.2 System Power Control

The power management unit logic provides two hardware means of controlling the CPU clock speed.

- *Doze mode* hardware causes the clock to the CPU core to be stopped and started in a periodic manner, resulting in power savings when there is no significant activity.
- *Thermal management* hardware forces the action to avoid overheating the CPU when the system is running at full speed for too long.

Both of these mechanisms engage the *stop clock* mechanism to slow down the CPU.

4.15.2.1 STPCLK# Mechanism to Control CPU Power Dissipation

The 3.3V Pentium processor contains a phase-locked loop (PLL) frequency generator that takes the external clock frequency input and multiplies it before applying it to the CPU core. The CPU core may be cut off from the PLL output by asserting the STPCLK# signal, without any loss of information. The CPU then enters the Stop Grant state in which the power consumption is approximately 20% of the normal consumption. It may be restarted almost immediately by negating the STPCLK# signal. Since a significant amount of power savings may be achieved when the CPU is in the Stop Grant state, it is forced into this state by FireStar when there is no significant activity.

On receiving an active STPCLK#, the CPU will generate a special bus cycle, and when it receives BRDY# from FireStar, it will enter the Stop Grant state. FireStar may be programmed so that a system interrupt, such as initiated by a keystroke or a timer interrupt, can restart the CPU almost immediately. Stopping the CPU clock is usually initiated by software, but could also be initiated by the hardware Doze mechanism.

The power consumed by the CPU can be controlled in two ways. The first method is to keep the STPCLK# signal asserted until a pre-programmed event causes FireStar to negate it. This could be used for maximum power saving modes invoked by software for prolonged periods of CPU inactivity.

The other method could be used for applications that do not require the CPU to operate at full speed all the time; the STPCLK# signal may be periodically asserted and negated. Since the Pentium processor does not provide much control for changing the frequency of the input clock, STPCLK# modulation is a viable alternative for saving power. This mode could be invoked by either software or hardware. On FireStar, the STPCLK# signal can be modulated by a base frequency of 32KHz, with a wide range of duty cycles. The cycle that is used to modulate the STPCLK# signal lasts 31.25µs. The time for which STPCLK# is asserted (low) is defined as t_{low}, and the time for which it is not asserted is defined as t_{hi}. The sum of t_{hi} and t_{low} equals 31.25µs. In every 32KHz cycle the STPCLK# signal is asserted for 31.25µs-t_{hi}. Different levels of power savings are obtained by programming the duration of t_{hi} through SYSCFG 41h[4:2].

Programming

To enable STPCLK# operation, the registers shown below must be programmed as follows.

- For maximum power savings set SYSCFG 66h[5] = 1 or for slow operation set SYSCFG 66h[5] = 0.
- Enable the STPCLK# logic mechanism by setting SYSCFG 61h[2] = 1.
- Enable the Stop Grant protocol by setting SYSCFG 66h[0] = 1 to recognize the Stop Grant cycle.

The STPCLK# sequence will now be observed any time the hardware Doze feature modulates the STPCLK# signal, or when APM commands the clock to stop. For example, during an APM operation for maximum power savings, FireStar:

1. Asserts its STPCLK# output
2. Waits for a Stop Grant cycle
3. Returns BRDY# to the CPU
4. Awaits a restart event (such as an interrupt)
5. Negates the STPCLK# signal, and continues operation.

For reduced power consumption, FireStar:

1. Asserts its STPCLK# output
2. Waits for a Stop Grant cycle
3. Returns BRDY# to the CPU
4. Waits for (31.25µs - t_{hi})
5. Negates the STPCLK# signal
6. Waits for t_{hi} (as programmed in SYSCFG 41h[4:2])
7. Goes back to Step 1.

While in Step 6, the CPU continues to execute instructions.

The registers associated with the STPCLK# feature are shown in Table 4-118.

Table 4-118 Register Bits Associated with STPCLK# Feature

7	6	5	4	3	2	1	0
SYSCFG 41h			**DOZE_TIMER Register 2**				**Default = 00h**
			Doze mode STPCLK# modulation (STPCLK# modulated by BCLK defined in SYSCFG E6h[7:6]): 000 = No Modulation (STPCLK# = 1) 001 = STPCLK# t_{hi} = 0.75 * 16 BCLKs 010 = STPCLK# t_{hi} = 0.5 * 16 BCLKs 011 = STPCLK# t_{hi} = 0.25 * 16 BCLKs 100 = STPCLK# t_{hi} = 0.125 * 16 BCLKs 101 = STPCLK# t_{hi} = 0.0625 * 16 BCLKs 110 = STPCLK# t_{hi} = 0.03125 * 32 BCLKs 111 = STPCLK# t_{hi} = 0.015625 * 64 BCLKs				
SYSCFG E6h if AEh[7] = 0			**Clock Source Register 4**				**Default = 70h**
BCLK source for STPCLK# modulation (For normal mode, Doze mode, thermal mgmt): 00 = 4KHz 01 = 32KHz (Default) 10 = 450KHz 11 = 900KHz							
SYSCFG 61h			**Debounce Register**				**Default = 00h**
					STPCLK# signal 0 = Disable 1 = Enable		
SYSCFG 65h			**Doze Register**				**Default = 00h**
			Recognize SMI during STPCLK#: 0 = No 1 = Yes				
SYSCFG 66h			**PMU Control Register 7**				**Default = 00h**
		Doze type: 0 = Modulate STPCLK# 1 = Keep STPCLK# asserted					STPGNT cycle wait option: 0 = Do not wait 1 = Wait for STPGNT cycle before negating STPCLK#

Table 4-118 Register Bits Associated with STPCLK# Feature (cont.)

7	6	5	4	3	2	1	0
SYSCFG 67h			**PMU Control Register 8**				**Default = 00h**
				Prevent STPCLK# generation by SYSCFG 50h[3] when INTR is active: 0 = Disable 1 = Enable	Normal mode STPCLK# modulation (read returns current STPCLK modulation setting only; STPCLK# modulated by BCLK defined in SYSCFG E6h[7:6]): 000 = No Modulation (STPCLK# = 1) 001 = STPCLK# t_{hi} = 0.75 * 16 BCLKs 010 = STPCLK# t_{hi} = 0.5 * 16 BCLKs 011 = STPCLK# t_{hi} = 0.25 * 16 BCLKs 100 = STPCLK# t_{hi} = 0.125 * 16 BCLKs 101 = STPCLK# t_{hi} = 0.0625 * 16 BCLKs 110 = STPCLK# t_{hi} = 0.03125 * 32 BCLKs 111 = STPCLK# t_{hi} = 0.015625 * 64 BCLKs		
SYSCFG 50h			**PMU Control Register 4**				**Default = 00h**
				Write = 1 to start Doze Read = Doze status: 0 = Counting 1 = Timed out			

4.15.3 Doze Mode

FireStar's power management unit includes Doze mode control logic. Doze is the state in which the CPU is fully alive and operational, yet running at a speed that is greatly reduced in order to save power. FireStar engages the Doze mode when it sees no activity in certain pre-definable areas for a certain time period. FireStar provides a choice of two time-out timers for each device. When an interrupt for the device or access in the range associated with that device occurs, the event triggers a Doze reset that reloads the selected timer for that device with the time-out value associated with that timer. Only when both time-outs have expired will the system return to Doze mode operation. Once initialized by software, the process is completely controlled by hardware. No further software intervention is needed, but an SMI can be generated if desired.

Even though the Doze mode is intended to operate autonomously without application or BIOS intervention, FireStar provides logic hooks to software for software-based power control. The most common type of software-based power control follows the Microsoft Advanced Power Management (APM) specification, which allows applications to inform the operating system when they are idle or do not require full processing power. The operating system, in turn, makes BIOS calls that can do any of the following:

- Turn off or put into a standby mode any unneeded peripherals
- Slow system clock speeds
- Turn off clocks to the CPU

Therefore, FireStar's power Doze mode logic provides for three Doze mode operations:

1) hardware-controlled slowdown,
2) software-controlled slowdown, and
3) software-controlled Doze with a stopped CPU clock (the clock to the CPU core is cut off).

The three modes are very similar and are outlined next, followed by register descriptions that the three modes have in common. FireStar provides the stop clock logic described next.

4.15.3.1 Dual Doze Timer Reload Selections

The standard DOZE_0 reload value can generate a Doze Timer time-out in as little as 2ms. However, this selection is not compatible with all applications. For example, stable operating speed might be desirable for at least 2sec after a keyboard interrupt in order to completely service the event ad prevent delays on subsequent keystrokes. Another operation, such as video access, might allow a return to Doze mode almost immediately.

Therefore, FireStar provides a choice of two time-out timers for each device. When an interrupt for the device or access in the range associated with that device occurs, the event triggers a Doze reset that reloads the selected timer for that device with the time-out value associated with that timer. Only when both timers have expired will the system return to Doze mode operation.

On earlier OPTi chipsets, COM port, LPT port, and GNR accesses could not cause a Doze reset. On FireStar these accesses can be programmed to enable a Doze reset. However, to maintain backward compatibility, the COM1, COM2, LPT, and GNR accesses point to the secondary doze timer at reset; this timer in turn is programmed for "no delay" at reset. The Doze logic interprets the "no delay" setting as inhibiting Doze reset for that source.

The SMI, EMPI1#, and INTR signals are also potential sources of Doze reset. However, these signals always use Doze Time-out 0 and cannot select Doze Time-out 1.

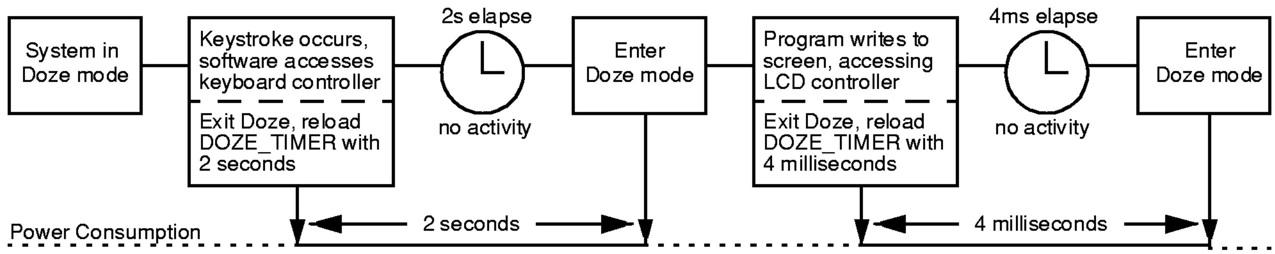
Doze reset events are all individually programmable to generate SMIs. In addition to the ability to reset the Doze mode when an SMI is encountered, FireStar has the ability to generate PMI#27 when the DOZE_TIMER times out. Doze time-out events on DOZE_0 and DOZE_1 can be individually programmed to generate an SMI through SYSCFG D9h[7:6] (shown in Table 4-119), which select the time-out(s) that will cause an SMI. Setting SYSCFG D9h[7:6] = 11 enables the PMI to generate an SMI.

Figure 4-39 shows an example of Dual Doze mode operation and Table 4-120 shows the setup and link registers associated with the example.

Table 4-119 SMI Generation on DOZE Time-Out

7	6	5	4	3	2	1	0
SYSCFG D9h			**PMU Event Register 6**				**Default = 00h**
DOZE_TIMER PMI#27 SMI: 00 = Disable 01 = Enable DOZE_0 10 = Enable DOZE_1 11 = Enable both							

Figure 4-39 Dual Doze Mode Operation Example



Setup: DOZE_0 = 2 seconds, SYSCFG 41h[7:5] = 110
DOZE_1 = 4 milliseconds, SYSCFG 79h[7:5] = 010
Link KBD_ACCESS to DOZE_0, SYSCFG 76h[6] = 0
Link LCD_ACCESS to DOZE_1, SYSCFG 76h[7] = 1

Table 4-120 Dual Doze Mode Operation Example Setup/Link Registers

7	6	5	4	3	2	1	0
SYSCFG 41h			**DOZE_TIMER Register 2**				**Default = 00h**
DOZE_0 time-out select: 000 = 2ms, 001 = 4ms, 010 = 8ms, 011 = 32ms, 100 = 128ms, 101 = 512ms, 110 = 2s, 111 = 8s Time-out generates PMI#27.							
SYSCFG 79h			**PMU Control Register 11**				**Default = 00h**
DOZE_1 time-out select: 000 = No delay (Default), 001 = 1ms, 010 = 4ms, 011 = 16ms, 100 = 64ms, 101 = 256ms, 110 = 1s, 111 = 4s							
SYSCFG 76h if AEh = 0			**Doze Reload Select Register 1**				**Default = 0Fh**
LCD_ ACCESS: 0 = DOZE_0 1 = DOZE_1	KBD_ ACCESS: 0 = DOZE_0 1 = DOZE_1						

4.15.3.2 Presetting Events to Reset Doze Mode

Before enabling Doze mode operation, whether hardware or software Doze mode, some preparation must be made for the event or events that will reset Doze mode and bring the system back to full operation. Otherwise, especially in the case of APM stop clock mode, there would be no way to execute CPU instructions to restart the CPU clock.

Therefore, it is first necessary to choose the source or sources that will perform a Doze reset. Doze reset will, if the system is currently in Doze mode, restore the system clocks to full operating speed. Doze reset also reloads the Doze timer with its originally programmed value.

- Setting SYSCFG 41h[1] = 1 enables LCD_ACCESS, KBD_ACCESS, DSK_ACCESS, HDU_ACCESS, GNR accesses, COM port accesses, and LPT accesses to reset Doze mode and reload the DOZE_TIMER. If the associated DOZE_TIMER has timed out and switched operation to Doze speed, this reload will change the system clocks back to their normal speed.
- SYSCFG 65h[5] selects the EPMI0# pin as a Doze reset trigger while bits [2:0] select EPMI[3:1]#, respectively.
- SYSCFG 62h[7:0], A2h[5:0], and 65h[3] define individual IRQs that can trigger a Doze reset.
- SYSCFG 65h[7] allows all enabled interrupts (i.e., any event that toggles the INTR signal to the CPU, to reset Doze mode).
- SYSCFG F6h[7:0] and F7h[7:0] are the DMA Doze Reload Registers. As stated previously Doze reset also reloads the Doze timer with its originally programmed value.

Once the Doze mode reset events have been programmed, either hardware or software Doze mode can be enabled.

Doze Reset Inside SMM

FireStar allows an SMI to reset Doze mode if STPCLK# is active from within System Management Mode. SYSCFG 65h[4] enables Doze mode reset when the SMI signal goes active, but since SMI is masked on entry to SMM, an SMI triggered while the system is in SMM is not seen until some other trigger resets Doze mode.

Setting SYSCFG 79h[4] = 1 handles Doze reset only from within SMM. Set SYSCFG 65h[4] = 1 to handle SMI Doze mode exit from outside of SMM.

Table 4-121 Register Bits that Select Doze Mode Reset Events

7	6	5	4	3	2	1	0
SYSCFG 41h			**DOZE_TIMER Register 2**				**Default = 00h**
						ACCESS events reset Doze mode: 0 = Disable 1 = Enable	
SYSCFG 62h			**IRQ Doze Register 1**				**Default = 00h**
IRQ13 Doze reset: 0 = Disable 1 = Enable	IRQ8 Doze reset: 0 = Disable 1 = Enable	IRQ7 Doze reset: 0 = Disable 1 = Enable	IRQ12 Doze reset: 0 = Disable 1 = Enable	IRQ5 Doze reset: 0 = Disable 1 = Enable	IRQ4 Doze reset: 0 = Disable 1 = Enable	IRQ3 Doze reset: 0 = Disable 1 = Enable	IRQ0 Doze reset: 0 = Disable 1 = Enable
SYSCFG 65h			**Doze Register**				**Default = 00h**
All interrupts to CPU reset Doze mode: 0 = Disable 1 = Enable	Reserved	EPMI0# Doze reset: 0 = Disable 1 = Enable	Recognize SMI during STPCLK#: 0 = No 1 = Yes	IRQ1 Doze reset: 0 = Disable 1 = Enable	EPMI3# Doze reset: 0 = Disable 1 = Enable	EPMI2# Doze reset: 0 = Disable 1 = Enable	EPMI1# Doze reset: 0 = Disable 1 = Enable

Table 4-121 Register Bits that Select Doze Mode Reset Events (cont.)

7	6	5	4	3	2	1	0
SYSCFG 79h			**PMU Control Register 11**				**Default = 00h**
			PMI# event triggers exit from Doze mode if the PMI event is enabled to generate SMI:(1) 0 = No 1 = Yes				
(1) For example, to let PMI#11 reset the Doze mode without generating SMI to the CPU, SYSCFG 5Ah[7:6] must = 11 and SYSCFG 5Bh[6] must = 1.							
SYSCFG A2h if AEh[7] = 0			**IRQ Doze Register 2**				**Default = 00h**
		IRQ15 Doze reset: 0 = Disable 1 = Enable	IRQ14 Doze reset: 0 = Disable 1 = Enable	IRQ11 Doze reset: 0 = Disable 1 = Enable	IRQ10 Doze reset: 0 = Disable 1 = Enable	IRQ9 Doze reset: 0 = Disable 1 = Enable	IRQ6 Doze reset: 0 = Disable 1 = Enable
SYSCFG F6h			**DMA Doze Reload Register 1**				**Default = 00h**
DRQ7 reloads DOZE_0: 0 = No 1 = Yes	DRQ6 reloads DOZE_0: 0 = No 1 = Yes	DRQ5 reloads DOZE_0: 0 = No 1 = Yes	IDE DDRQ reloads DOZE_0:(1) 0 = No 1 = Yes	DRQ3 reloads DOZE_0: 0 = No 1 = Yes	DRQ2 reloads DOZE_0: 0 = No 1 = Yes	DRQ1 reloads DOZE_0: 0 = No 1 = Yes	DRQ0 reloads DOZE_0: 0 = No 1 = Yes
(1) Bit 4 controls whether the DDRQ line from bus mastering IDE drives can reload the timers. The bit controls DDRQ from both cables, so enabling the reload feature on any one bus mastering drive enables it for all present.							
SYSCFG F7h			**DMA Doze Reload Register 2**				**Default = 00h**
DRQ7 reloads DOZE_1: 0 = No 1 = Yes	DRQ6 reloads DOZE_1: 0 = No 1 = Yes	DRQ5 reloads DOZE_1: 0 = No 1 = Yes	IDE DDRQ reloads DOZE_1:(1) 0 = No 1 = Yes	DRQ3 reloads DOZE_1: 0 = No 1 = Yes	DRQ2 reloads DOZE_1: 0 = No 1 = Yes	DRQ1 reloads DOZE_1: 0 = No 1 = Yes	DRQ0 reloads DOZE_1: 0 = No 1 = Yes
(1) Bit 4 controls whether the DDRQ line from bus mastering IDE drives can reload the timers. The bit controls DDRQ from both cables, so enabling the reload feature on any one bus mastering drive enables it for all present.							

DEVSEL# Doze Reset

Activity on the PCI bus can reset the Doze mode and cause a return to full operating speed. FireStar's logic provides two bits to enable Doze reset separately for PCI I/O accesses and memory accesses. The Doze reset is triggered by DEVSEL# going active and is qualified by the M/IO# signal.

Table 4-122 PCI Bus Doze Reset Registers

7	6	5	4	3	2	1	0
SYSCFG A2h if AEh[7] = 0			**IRQ Doze Register 2**				**Default = 00h**
PCI bus I/O access Doze reset: 0 = Disable 1 = Enable	PCI memory access Doze reset: 0 = Disable 1 = Enable						
SYSCFG A2h if AEh[7] = 1			**IRQ Doze Register 2**				**Default = 00h**
PREQ# Doze reset: 0 = Disable 1 = Enable	CLKRUN# Doze reset: 0 = Disable 1 = Enable	Reserved					
SYSCFG 78h			**Doze Reload Select Register 3**				**Default = 00h**
PCI: 0 = DOZE_0 1 = DOZE_1							

4.15.3.3 Automatic (Hardware) Doze Mode

The chipset can be set up for hardware-controlled slowdown Doze mode by programming the following information.

1. Set up the hardware and programming to consider the stop clock mechanism as described in Section 4.15.2.1, "STPCLK# Mechanism to Control CPU Power Dissipation" on page 187.
2. Program the events that will reset Doze mode as described in "Presetting Events to Reset Doze Mode" on page 192.
3. Associate the access event with their time-out counters (DOZE_0 or DOZE_1), by appropriately programming SYSCFG 76h through 78h.
4. Select the desired time-outs, that is, the time required after the last event before the system can be considered "inactive," in SYSCFG 41h[7:5] and in 79h[7:5]. A two second (2s) time-out is typical.
5. Select the STPCLK# duty cycle from SYSCFG 41h[4:2]. Time t_{hi} is defined as that time for which STPCLK# is not asserted in one 32KHz period.
6. Set SYSCFG 66h[5] = 0 for slowdown.
7. Set SYSCFG 65h[4] = 1 if the chipset should exit Doze mode for SMIs, or = 0 if the SMIs can run adequately at the Doze speed.
8. Finally, enable the hardware DOZE_TIMER by setting SYSCFG 41h[0] = 0.

After the selected period of inactivity, the hardware Doze mode is entered and the STPCLK# signal is modulated. In the event of any of the enabled accesses, SMIs, or IRQs, the CPU is switched back to full speed operation, and the Doze timer associated with the access event is reloaded.

4.15.3.4 APM (Software) Doze Mode

FireStar can be set up for software-initiated, reduced CPU power consumption or maximum power saving mode in a very straightforward manner.

1. Set up the hardware and programming to consider the stop clock mechanism as described in the Section 4.15.2.1, "STPCLK# Mechanism to Control CPU Power Dissipation" on page 187.
2. Program the events that will reset the Doze mode as described in the "Presetting Events to Reset Doze Mode" on page 192.
3. Set SYSCFG 65h[4] = 1 if FireStar should exit Doze mode for SMIs, or = 0 if the SMIs can run adequately at the Doze speed. Obviously, SYSCFG 65h[4] must be set to 1 if maximum power savings is desired, or else the SMI will be missed altogether.
4. Select the duty cycle for STPCLK# from SYSCFG 41h[4:2].
5. Disable the hardware DOZE_TIMER by setting SYSCFG 41h[0] = 1.

At this point the system is ready for APM control. When APM makes a call for low or very low power operation, the BIOS or power management code simply:

- Sets SYSCFG 66h[5] = 0 for reduced CPU power consumption or sets SYSCFG 66h[5] = 1 for maximum power savings mode
- Sets SYSCFG 50h[3] = 1 to initiate the Doze mode

On the event of any of the enabled ACCESSes, SMIs, or IRQs, the clock to the CPU core will be enabled all the time, until the above two steps are repeated again.

Start Doze Bit

SYSCFG 50h[3] serves two purposes: to start the Doze mode and to read the DOZE_TIMER status.

- Write: Start APM Doze mode
  1 = Start Doze mode (if SYSCFG 40h[0] = 1)
  0 = No effect
- Read: Hardware DOZE_TIMER time-out status bit
  1 = Hardware DOZE_TIMER has timed out
  0 = Hardware DOZE_TIMER still counting

Table 4-123 shows the hardware and software related Doze mode registers.

Table 4-123 Hardware and Software Doze Mode Registers

7	6	5	4	3	2	1	0
SYSCFG 76h if AEh = 0			**Doze Reload Select Register 1**				**Default = 0Fh**
LCD_ ACCESS: 0 = DOZE_0 1 = DOZE_1	KBD_ ACCESS: 0 = DOZE_0 1 = DOZE_1	DSK_ ACCESS: 0 = DOZE_0 1 = DOZE_1	HDU_ ACCESS: 0 = DOZE_0 1 = DOZE_1	COM1&2_ ACCESS: 0 = DOZE_0 1 = DOZE_1	LPT_ ACCESS: 0 = DOZE_0 1 = DOZE_1	GNR1_ ACCESS: 0 = DOZE_0 1 = DOZE_1	GNR2_ ACCESS: 0 = DOZE_0 1 = DOZE_1
SYSCFG 76h if AEh = 1			**Doze Reload Select Register 1**				**Default = 03h**
Reserved		PREQ#: 0 = DOZE_0 1 = DOZE_1	CLKRUN#: 0 = DOZE_0 1 = DOZE_1	Reserved		GNR5: 0 = DOZE_0 1 = DOZE_1	GNR6: 0 = DOZE_0 1 = DOZE_1
SYSCFG 77h			**Doze Reload Select Register 2**				**Default = 00h**
IRQ8: 0 = DOZE_0 1 = DOZE_1	IRQ7: 0 = DOZE_0 1 = DOZE_1	IRQ6: 0 = DOZE_0 1 = DOZE_1	IRQ5: 0 = DOZE_0 1 = DOZE_1	IRQ4: 0 = DOZE_0 1 = DOZE_1	IRQ3: 0 = DOZE_0 1 = DOZE_1	IRQ1: 0 = DOZE_0 1 = DOZE_1	IRQ0: 0 = DOZE_0 1 = DOZE_1
SYSCFG 78h			**Doze Reload Select Register 3**				**Default = 00h**
PCI: 0 = DOZE_0 1 = DOZE_1	IRQ15: 0 = DOZE_0 1 = DOZE_1	IRQ14: 0 = DOZE_0 1 = DOZE_1	IRQ13: 0 = DOZE_0 1 = DOZE_1	IRQ12: 0 = DOZE_0 1 = DOZE_1	IRQ11: 0 = DOZE_0 1 = DOZE_1	IRQ10: 0 = DOZE_0 1 = DOZE_1	IRQ9: 0 = DOZE_0 1 = DOZE_1
SYSCFG 41h			**DOZE_TIMER Register 2**				**Default = 00h**
DOZE_0 time-out select: 000 = 2ms 001 = 4ms 010 = 8ms 011 = 32ms 100 = 128ms 101 = 512ms 110 = 2s 111 = 8s Time-out generates PMI#27.			Doze mode STPCLK# modulation (STPCLK# modulated by BCLK defined in SYSCFG E6h[7:6]): 000 = No Modulation (STPCLK# = 1) 001 = STPCLK# t_{hi} = 0.75 * 16 BCLKs 010 = STPCLK# t_{hi} = 0.5 * 16 BCLKs 011 = STPCLK# t_{hi} = 0.25 * 16 BCLKs 100 = STPCLK# t_{hi} = 0.125 * 16 BCLKs 101 = STPCLK# t_{hi} = 0.0625 * 16 BCLKs 110 = STPCLK# t_{hi} = 0.03125 * 32 BCLKs 111 = STPCLK# t_{hi} = 0.015625 * 64 BCLKs				Doze control select: 0 = Hardware 1 = Software
SYSCFG 79h			**PMU Control Register 11**				**Default = 00h**
DOZE_1 time-out select: 000 = No delay (Default) 001 = 1ms 010 = 4ms 011 = 16ms 100 = 64ms 101 = 256ms 110 = 1s 111 = 4s							
SYSCFG 66h			**PMU Control Register 7**				**Default = 00h**
		Doze type: 0 = Modulate STPCLK# 1 = Keep STPCLK# asserted					

Table 4-123 Hardware and Software Doze Mode Registers (cont.)

7	6	5	4	3	2	1	0
SYSCFG 65h			**Doze Register**				**Default = 00h**
			Recognize SMI during STPCLK#: 0 = No 1 = Yes				
SYSCFG 50h			**PMU Control Register 4**				**Default = 00h**
				Write = 1 to start Doze Read = Doze status: 0 = Counting 1 = Timed out			

4.15.4 CPU Thermal Management Unit

Thermal management hardware is implemented in FireStar for monitoring the level of CPU activity and the operating temperature of the device. A flexible hardware scheme monitors CPU temperature to determine when it is necessary to enter cool-down clocking mode. In this way, a serious over-temperature condition cannot get out of control.

4.15.4.1 Operating Temperature Ranges

The FireStar thermal management algorithm identifies the temperature limits of the CPU through activity level values that correspond with idle, equilibrium, and thermal runaway conditions.

Idle Condition

The CPU is cool to the touch. FireStar is using APM stop clock or hardware Doze mode to save power, and no real activity is taking place. This operational level constitutes the base level of activity, so it does not require a register to hold the value. It is associated in the FireStar thermal management scheme with “zero”.

Equilibrium Condition

The CPU is warm to the touch. It is operating at full capacity for short bursts but frequently is in low power modes. The heat generated by the CPU is dissipated at the same rate that it is produced. Most active computer usage falls into this category, where APM or hardware Doze mode operates frequently enough to allow safe operation. The FireStar thermal management scheme associates this temperature with LOFREQ[15:0]. The value of these bits is referred to as LOFREQ throughout this section.

Thermal Runaway Condition

The CPU is hot to the touch. It is running at its full rated speed and generating heat faster than it can be dissipated, so its temperature increases. The program being used is active enough that APM or hardware Doze mode cannot operate often enough to hold the temperature down. Operating in this mode for an extended period may cause the CPU to fail. The FireStar thermal management scheme associates this temperature with HIFREQ[15:0]. The value of these bits is referred to as HIFREQ throughout this section.

Cool-down Clocking

Cool-down clocking takes place according to the reduced clock rate programmed into the Cool-down Clock Rate bits SYSCFG A5h[5:3] and A5h[2:0]. FireStar keeps the CPU running at the cool-down clocking rate speed for the Cool-down Holdoff period.

4.15.4.2 Fail-Safe Thermal Management

The thermal management unit accepts a periodic waveform that varies with temperature as input from an inexpensive external sensor circuit. The goal is to provide actual temperature sensing and monitoring without resorting to expensive sensor devices or on-chip analog circuitry.

On-board logic monitors the waveform, a low-frequency (100Hz-10kHz) square or sine wave, and can determine actual CPU temperature by the frequency of the waveform. The hardware reacts to changes in this frequency according to programmable parameters. Should the external circuit fail to generate the expected waveform, the fail-safe logic automatically maintains cool-down clocking mode to the CPU for an indefinite period.

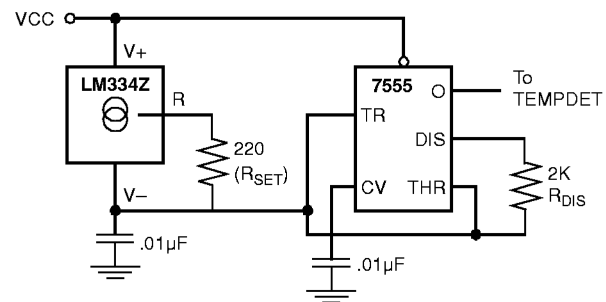
In addition, the hardware keeps track of the frequency and allows software to read the value at any time. In this way, SMM code will be able to return an actual calibrated CPU temperature to system management programs.

Suggested Circuit

A relatively simple and inexpensive circuit can be used to measure CPU temperature. The suggested circuit is based on the LM334Z three-terminal adjustable current source, available from National Semiconductor. The sense voltage of this device is directly proportional to absolute temperature. The circuit also uses a 7555 timer to convert the voltage to a proportional frequency.

Figure 4-40 illustrates the circuit recommended to generate a frequency that is directly proportional to temperature. Only the LM334Z part is sensitive to temperature, and must be mounted under or on the CPU (or in an appropriate heat sink location).

Figure 4-40 External Temperature Sensor Circuit



The following equation describes the output frequency of this circuit.

$$f = \frac{3}{C} \times \left\{ \frac{227uV \times (273 + T_C)}{(V_{CC} \times R_{SET}) + [2.085) \times R_{DIS} \times 227uV \times (273 + T_C)]} \right\}$$

If C is chosen as 0.01uF, R_{SET} as 220 ohms, R_{DIS} as 2K ohms, and VCC = 5.0V, then the equation is simply:

$$f = \frac{300 \times [61971 + (227 \times T_C)]}{1358.4 + (0.95 \times T_C)}$$

Registers

All registers must be programmed before the thermal management THKEN enable bit (SYSCFG A5h[7]) is set. Refer to Table 4-124 for information regarding the thermal management registers.

Operation

The fail-safe thermal management unit will compare the THKFREQ value with the low frequency limit (LOFREQ) and high frequency limit (HIFREQ) values every second to determine when to engage thermal management. When thermal management is engaged, STPCLK modulation duty cycle will be based on L1THK or L2THK values. SMI may be generated as well.

There are three frequency ranges that THKFREQ falls into:

1) 0 = < THKFREQ < LOFREQ
2) LOFREQ = < THKFREQ < HIFREQ
3 HIFREQ = < THKFREQ < 64 KHz

Case A: L2_HIFREQ = 0, THKFREQ decreases as temp increases.

> IF (THKFREQ = range #3)
> THEN temp is not high enough -- STPCLK modulation = none;
>
> ELSE IF (THKFREQ = range #2)
> THEN temp is high -- STPCLK modulation = L1THK duty cycle;
>
> ELSE
> temp is very high -- STPCLK modulation = L2THK duty cycle;
>
> END IF;

Case B: L2_HIFREQ = 1, THKFREQ increases as temp increases.

> IF (THKFREQ = range #1)
> THEN temp is not high enough -- STPCLK modulation = none;
>
> ELSE IF (THKFREQ = range #2)
> THEN temp is high -- STPCLK modulation = L1THK duty cycle;
>
> ELSE temp is very high -- STPCLK modulation = L2THK duty cycle;
>
> END IF;

In both cases, when L1THK or L2THK STPCLK modulation is engaged, a register bit will be set to indicate thermal management is engaged. SMI may also be generated as an option.

The emergency overtemp sense pin will still override all the Doze mode and fail-safe stop clock modulation.

Programming

Before programming thermal management, the STPCLK# mechanism must be set up for the CPU being used as described in Section 4.15.2.1, "STPCLK# Mechanism to Control CPU Power Dissipation". Enabling thermal management locks the STPCLK# bits and prevents them from being altered until the next hardware reset.

The thermal management option must be configured by setting the bits in SYSCFG A5h-AAh (refer to Table 4-124), and then setting SYSCFG A5h[7] = 1. The register setting order is not important, but bit 7 of SYSCFG A5h must be set to 1 only after all other settings have been made. Once bit 7 is set, none of the thermal management registers can be written again without resetting FireStar.

If the Doze mode is in progress when the thermal management unit engages (or vice-versa), the lower value of t_{hi} as set by the two schemes for STPCLK# modulation will be used.

Table 4-124 Thermal Management Registers

7	6	5	4	3	2	1	0
SYSCFG A5h			**Thermal Management Register 1**				**Default = 00h**
Thermal Mgmt.: 0 = Disable 1 = Enable	TEMPDET Variation - As temperature increases, frequency: 0 = Decreases 1 = Increases	Level 2 STPCLK# modulation rate - Sets the stop clock throttling rate when temperature enters second (overtemp) range: 000 = No modulation (STPCLK# = 1) 001 = STPCLK# t_{hi} = 0.75 * 16 BCLKs 010 = STPCLK# t_{hi} = 0.5 * 16 BCLKs 011 = STPCLK# t_{hi} = 0.25 * 16 BCLKs 100 = STPCLK# t_{hi} = 0.125 * 16 BCLKs 101 = STPCLK# t_{hi} = 0.0625 * 16 BCLKs 110 = STPCLK# t_{hi} = 0.03125 * 32 BCLKs 111 = STPCLK# t_{hi} = 0.015625 * 64 BCLKs			Level 1 STPCLK# modulation rate - Sets the stop clock throttling rate when temperature enters first (high temp) range: 000 = No modulation (STPCLK# = 1) 001 = STPCLK# t_{hi} = 0.75 * 16 BCLKs 010 = STPCLK# t_{hi} = 0.5 * 16 BCLKs 011 = STPCLK# t_{hi} = 0.25 * 16 BCLKs 100 = STPCLK# t_{hi} = 0.125 * 16 BCLKs 101 = STPCLK# t_{hi} = 0.0625 * 16 BCLKs 110 = STPCLK# t_{hi} = 0.03125 * 32 BCLKs 111 = STPCLK# t_{hi} = 0.015625 * 64 BCLKs		
Note: Once thermal management has been enabled (bit 7 = 1), none of the thermal management registers can be overwritten.							

Table 4-124 Thermal Management Registers

7	6	5	4	3	2	1	0
SYSCFG A6h			**Thermal Management Register 2**				**Default = 00h**
LOFREQ[7:0]: Low frequency limit low byte							
SYSCFG A7h			**Thermal Management Register 3**				**Default = 00h**
LOFREQ[15:8]: Low frequency limit high byte							
SYSCFG A8h			**Thermal Management Register 4**				**Default = 00h**
HIFREQ[7:0]: High frequency limit low byte							
SYSCFG A9h			**Thermal Management Register 5**				**Default = 00h**
HIFREQ[15:8]: High frequency limit high byte							
SYSCFG AAh			**Thermal Management Register 6**				**Default = 00h**
Emergency Overtemp Sensor STPCLK# Modulation Rate: 000 = No modulation (STPCLK# = 1) 001 = STPCLK# t_{hi} = 0.75 * 16 BCLKs 010 = STPCLK# t_{hi} = 0.5 * 16 BCLKs 011 = STPCLK# t_{hi} = 0.25 * 16 BCLKs 100 = STPCLK# t_{hi} = 0.125 * 16 BCLKs 101 = STPCLK# t_{hi} = 0.0625 * 16 BCLKs 110 = STPCLK# t_{hi} = 0.03125 * 32 BCLKs 111 = STPCLK# t_{hi} = 0.015625 * 64 BCLKs			THMIN pin polarity: 0 = High 1 = Low	EPMI trigger for thermal mgmt. 00 = EPMI0# 01 = EPMI1# 10 = EPMI2# 11 = EPMI3# Also see bit 1.		THMIN input: 0 = THMIN 1 = EPMI indicated in bits [3:2]	HDI input: 0 = HDI 1 = EPMI indicated by SYSCFG F0h[1:0]
SYSCFG F3h			**Thermal Management Register 7**				**Default = 00h**
THFREQ[7:0] - Current frequency low byte: - THFREQ[15:0] return a value from 0 to 65535. This value is updated once per second so that software can read the input pin frequency in kHz and correlate the value to the actual CPU temperature at that moment.							
SYSCFG F4h			**Thermal Management Register 8**				**Default = 00h**
THFREQ[15:8] - Current frequency high byte							

4.15.4.3 SMI Generation

When the thermal management unit engages or disengages cool-down clocking, an SMI on PMI#25 can be generated. This feature is controlled through SYSCFG DBh[6]. When this SMI is being serviced, power management code can read SYSCFG A5h[7] to determine whether it was an entry into or an exit from cool-down clocking mode that caused the SMI.

PMI#25 is also shared with the EPMI3# event. If the SMI from EPMI3# is also enabled, on entry to the SMI software must check the state of the EPMI3# signal to determine whether the reason for the SMI is the external EPMI3# event or cool-down clocking entry/exit.

Emergency Overtemp Sense

It is possible for an external sensor to force FireStar permanently into cool-down clocking mode according to the parameters programmed for thermal management. When low, the EPMIx# input (any EPMI can be assigned at SYSCFG AAh[3:2]) can cause FireStar to enter cool-down clocking mode. The existing SMI enable bits for EPMIx# are still operational regardless of the setting of the emergency overtemp enable bit (SYSCFG A1h[2]). Therefore, an overtemp condition can also be programmed to cause an SMI so that the power management firmware will be made aware of the situation and instruct the user to shut down the system. In this way, a serious over-temperature condition cannot get out of control.

The chip will remain in cool-down clocking mode, using the rate specified in Thermal Management Register 3 (defaults to a 50% duty cycle for STPCLK#), as long as the EPMIx# input remains triggered. The trigger specifications are the same as those for triggering the SMI: SYSCFG 40h[2:1] for EPMI[1:0]# and SYSCFG DBh[5:4] for EPMI[3:2]# select whether a high- or a low-level is active, and this same bit selects whether a high- or a low-level will engage cool-down clocking. The thermal management unit does **not** need to be enabled to use this feature.

Programming

This option is enabled by writing SYSCFG A1h[2]. Once written, this bit **cannot** be changed without a hard reset of the chip.

When SYSCFG DBh[6] = 1, entry into or exit from cool-down clocking mode causes PMI#25 and an SMI. If the EPMIx# event is also programmed to cause an SMI, the following situation can occur.

1. The thermal sensor input changes state, causing PMI#2 and an SMI.
2. Power management code services the SMI.
3. The thermal management unit enters cool-down clocking mode.
4. At this point, PMI#25 is generated along with *another* SMI.

Therefore, two SMIs will have been generated. On exit from cool-down clocking mode, only one SMI will be generated, since the active-to-inactive transition on EPMIx# does not cause another SMI. Power management software must be able to anticipate this situation and deal with it appropriately.

Table 4-125 shows the above discussed register bits.

Table 4-125 SMI Generation on Cool-down Clocking Entry/Exit Registers

7	6	5	4	3	2	1	0
SYSCFG 40h			**PMU Control Register 1**				**Default = 00h**
					EPMI1# polarity: 0 = Active high 1 = Active low	EPMI0# polarity: 0 = Active high 1 = Active low	
SYSCFG A1h			**Feature Control Register 2**				**Default = 00h**
					Emerg. over-temp sense: 0 = Disable 1 = Enable		
SYSCFG A5h			**Thermal Management Register 1**				**Default = 00h**
Thermal Mgmt.: 0 = Disable 1 = Enable							
Note: Once thermal management has been enabled (bit 7 = 1), none of the thermal management registers can be overwritten.							

Table 4-125 SMI Generation on Cool-down Clocking Entry/Exit Registers (cont.)

7	6	5	4	3	2	1	0
SYSCFG AAh			**Thermal Management Register 6**				**Default = 00h**
			THMIN pin polarity: 0 = High 1 = Low	EPMI trigger for thermal mgmt. 00 = EPMI0# 01 = EPMI1# 10 = EPMI2# 11 = EPMI3# Also see bit 1.		THMIN input: 0 = THMIN 1 = EPMI indicated in bits [3:2]	HDI input: 0 = HDI 1 = EPMI indicated by SYSCFG F0h[1:0]
SYSCFG DBh if AEh[7] = 0			**Next Access Event Generation Register 1**				**Default = 00h**
	SMI on cool-down clocking entry/exit: 0 = Disable 1 = Enable	EPMI3# pin polarity: 0 = Active high 1 = Active low	EPMI2# pin polarity: 0 = Active high 1 = Active low				
SYSCFG F0h			**Hot Docking Control Register 2**				**Default = 00h**
						EPMI trigger for HDI: 00 = EPMI0# 01 = EPMI1# 10 = EPMI2# 11 = EPMI3# Also see SYSCFG AAh[0]	

4.15.5 Suspend and Resume

FireStar offers the ability to halt operations at extremely low power yet retain all its programming, called Suspend. FireStar will respond to interrupts to determine that a return to normal operation, called Resume, is necessary.

4.15.5.1 Suspend Mode

Suspend mode provides a significant level of power conservation. The Suspend initiation event, either a key or button depression or a time-out SMI, calls a software routine in SMM code to save the current state of the system for complete restoration at some later time. In this mode, most system power can be shut down while still retaining restorability. The lowest power consumption mode is that in which the DRAM is kept powered up (the system state is stored in the DRAM), the CPU is powered off, and the cache is also powered off. FireStar enters the Suspend mode when SYSCFG 50h[0] is set to 1. Software must control this event, even though a timer time-out may have initiated the process, because CPU processing must be completed in an orderly manner. Upon resuming from the Suspend mode, the controlling code **must** clear the Suspend PMI Event, PMI#7, by writing SYSCFG 5Ch[7] = 1. Otherwise, FireStar will never exit the Suspend mode the next time SYSCFG 50h[0] is set to 1.

The registers shown in Table 4-126 select the state of various signals during the Suspend mode. SYSCFG 59h[6] is used so that the system timers can be restarted to prevent false time-outs upon resuming from the Suspend mode. Refer to the Section 4.15.8, "Resume Event", to determine how to select the events that will cause FireStar to Resume operation after Suspend.

Table 4-126 Suspend Control Register Bits

7	6	5	4	3	2	1	0
SYSCFG 50h			**PMU Control Register 4**				**Default = 00h**
							Start Suspend (WO): 1 = Enter Suspend mode
SYSCFG 59h			**PMU Event Register 2**				**Default = 00h**
	Reload timers on Resume: 0 = No 1 = Yes						
SYSCFG 5Ch			**PMI SMI Source Register 1 (Write 1 to Clear)**				**Default = 00h**
PMI#7, Suspend: 0 = Not Active 1 = Active							
SYSCFG ADh			**Feature Control Register 3**				**Default = 00h**
		CPU power state in Suspend: 0 = Powered 1 = 0 Volt					

Suspend Mode Power Savings

While in the Suspend mode, FireStar can be programmed to save power in two ways:

1) short-pulse refresh for normal DRAM or
2) self-refresh for self-refresh DRAM, and slow interrupt scan.

Table 4-127 shows the registers/bits associated with Suspend mode power savings.

Suspend Mode Refresh

The Suspend refresh rate is selected by programming the SYSCFG 12h[5:4] and/or 12h[3:2]. During the Suspend mode, refresh can be:

1. slow refresh based on the REFRESH#/32KHZ input, with a wide RAS# pulse width,
2. self-refresh initiated by 32KHz, or
3. short pulse width refresh based on the 32KHz clock.

The first option generates refresh in the same manner that refresh is generated during normal mode, with the RAS pulse width determined by SYSCFG 01h[5:4].

The second option can be used for self-refresh DRAMs with the initial control required to put the DRAM in self-refresh mode based on the 32KHz clock. The short pulse refresh mechanism is engaged for lowest power consumption during Suspend for non self-refresh DRAM. In this mode, the refresh is based on the 32KHz clock, but the refresh pulse width is 100ns only. This pulse width is capable of retaining the DRAM data and results in lowest power consumption.

Multiplexed Inputs Scan Rate

The logic can scan for multiplexed inputs to FireStar interrupts at a slower rate during refresh. During normal mode, the sample rate is governed by the internally generated ATCLK. During Suspend mode, SYSCFG 66h[6] controls the frequency of ATCLK, and therefore, the sampling frequency. Note that a major reduction of power consumption can come from switching off all the clocks except the 32KHz clock during Suspend.

Table 4-127 Suspend Mode Power Saving Feature Related Bits

7	6	5	4	3	2	1	0
SYSCFG 01h			**DRAM Control Register 1**				**Default = 00h**
		RAS pulse width used during refresh: 00 = 7 CPUCLKs 01 = 6 CPUCLKs 10 = 5 CPUCLKs 11 = 4 CPUCLKs					
SYSCFG 12h			**Refresh Control Register**				**Default = 00h**
		Suspend mode refresh: 00 = From CPUCLK state machine 01 = Self-refresh based on 32KHz only 10 = Normal refresh based on 32KHz only 11 = Reserved		Slow refresh: Refresh on: 00 = Every REFRESH#/32KHz falling edge 01 = Alternate REFRESH#/32KHz falling edge 10 = One in four REFRESH#/ 32KHz falling edge 11 = Every REFRESH#/32KHz toggle			

Table 4-127 Suspend Mode Power Saving Feature Related Bits

7	6	5	4	3	2	1	0
SYSCFG 66h			**PMU Control Register 7**				**Default = 00h**
Suspend-to-Normal refresh delay: 0 = None 1 = Three 32KHz CLKs Write to 1 always.	Suspend mode ATCLK frequency: 0 = Derived from PCICLK 1 = 32KHz Setting can be overridden by SYSCFG 79h[0]						
SYSCFG 79h			**PMU Control Register 8**				**Default = 00h**
							ATCLK during Suspend: 0 = Run 1 = Stopped (overrides SYSCFG 66h[6])

4.15.6 Chip-Level Power Conservation Features

A central design goal of FireStar was to incorporate power-reducing features wherever possible. To this end, several innovative methods of power conservation are implemented.

4.15.6.1 Leakage Control

FireStar offers leakage control features to reduce Suspend mode power consumption. Each pin (or pin group in the case of certain bus signals) can be set as follows during Suspend mode:

- Tristated only
- Tristated and pulled down
- Tristated and pulled up
- Maintained (driven) at previous value
- Driven low

These options allow pins to be properly managed in all designs. The register format is shown Table 4-128. Leakage control registers start at PCIDV1 70h.

Table 4-128 Leakage Control Registers

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">PCIDV1 70h</td><td colspan="4">Leakage Control Register - Byte 0</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2">W/R#, HITM#, FERR#, SMIACT#
Suspend state:
00 = No pull-downs
01 = Pull-down
10 = Reserved
11 = Reserved</td><td colspan="2">BE[7:0]#, M/IO#, D/C#, CACHE#, LOCK# Suspend state:
00 = No pull-downs
01 = Pull-down during BOFF#
10 = Pull-down during Suspend
11 = Pull-down during BOFF# and Suspend</td><td colspan="2">HD[63:0]
Suspend and Idle state:
00 = Tristate
01 = Tristate, pull-down
10 = Reserved
11 = Reserved</td><td colspan="2">HA[31:3]
Suspend state:
00 = Tristate
01 = Tristate, pull-down
10 = Reserved
11 = Reserved</td></tr>
<tr><td colspan="2">PCIDV1 71h</td><td colspan="4">Leakage Control Register - Byte 1</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2">IGERR#, A20M#
Suspend state:
00 = Drive active
01 = Drive inactive
10 = Tristate, pull-down
11 = Reserved</td><td colspan="2">CPURST, CPUINIT, AHOLD, NMI, INTR, STPCLK#
Suspend state:
00 = Drive
01 = Tristate
10 = Reserved
11 = Reserved</td><td>BRDY#, NA#, KEN#, EADS#, BOFF#, SMI#
Suspend state:
0 = Drive
1 = Tristate</td><td colspan="3">MD[63:0]
Suspend state:
XX1 = Pull-down at Idle
X1X = Pull-down in STPCLK#
1XX = Pull-down in Suspend</td></tr>
<tr><td colspan="2">PCIDV1 72h</td><td colspan="4">Leakage Control Register - Byte 2</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2">REQ3#, GNT3#
Suspend state:
00 = Drive
01 = Tristate
10 = Reserved
11 = Reserved</td><td colspan="2">C/BE[3:0]#, IRDY#, TRDY#, STOP#, AD[31:0], LOCK#, FRAME#, PAR, PERR#, SERR#, DEVSEL#, GNT1#, REQ0#, GNT0#, REQ2#, GNT2#
Suspend state:
00 = Drive
01 = Tristate
10 = Tristate, pull-down
11 = Reserved</td><td colspan="2">TAG[7:0] state:
X1 = Tristate pull-down in STPCLK#
1X = Tristate pull-down in Suspend</td><td>BWE#, GWE#
Suspend state:
0 = Drive
1 = Tristate</td><td>CACS#
Suspend state:
0 = Drive
1 = Tristate, pull-down</td></tr>
</table>

Table 4-128 Leakage Control Registers (cont.)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">PCIDV1 73h</td><td colspan="4">Leakage Control Register - Byte 3</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2">CMD#
Suspend state:
00 = Drive
01 = Tristate
10 = Tristate, pull-down
11 = Reserved</td><td colspan="2">SD[15:0]
Suspend state:
01 = No pull-up/down in Active mode, tristate in Suspend mode
01 = Pull-up in Active mode, tristate in Suspend mode
10 = No pull-up/down in Active mode, pull-down in Suspend mode
11 = Pull-up in Active mode, pull-down in Suspend mode</td><td colspan="2">DBEW#
Suspend state:
00 = Drive
01 = Tristate
10 = Tristate, pull-down
11 = Reserved</td><td colspan="2">IRQSER
Suspend state:
00 = Drive
01 = Tristate
10 = Tristate, pull-down
11 = Reserved</td></tr>
<tr><td colspan="2">PCIDV1 74h</td><td colspan="4">Leakage Control Register - Byte 4</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2">SA[15:0]
Suspend state:
00 = Drive
01 = Tristate
10 = Tristate, pull-down
11 = Reserved</td><td colspan="2">SA[23:18]
Suspend state:
00 = Drive
01 = Tristate
10 = Tristate, pull-down
11 = Reserved</td><td colspan="2">XD[7:0]
Suspend state:
01 = No pull-up/down in Active mode, tristate in Suspend mode
01 = Pull-up in Active mode, tristate in Suspend mode
10 = No pull-up/down in Active mode, pull-down in Suspend mode
11 = Pull-up in Active mode, pull-down in Suspend mode</td><td colspan="2">RFSH#, MRD#, MWR#, IORD#, IOWR#
Suspend state:
00 = Drive
01 = Tristate
10 = Tristate, pull-down
11 = Reserved</td></tr>
<tr><td colspan="2">PCIDV1 75h</td><td colspan="4">Leakage Control Register - Byte 5</td><td colspan="2">Default = 00h</td></tr>
<tr><td>Secondary IDE interface in ISA-less mode:
0 = Tristated
1 = Driven</td><td>XD bus (primary IDE interface) in no XD bus mode:
0 = Tristated
1 = Driven
This bit must be set if the RTCAS:A20M# strap option = 10 to allow IDE control signals to be driven on the XD bus.</td><td>MD[63:0] engage pull-down:
0 = Controlled by PCIDV1 71[2:0]h
1 = Pull-down always (overrides PCIDV1 71h[2:0])</td><td>TAG[7:0] engage pull-down:
0 = Controlled by PCIDV1 72h[3:2]
1 = Pull-down always (overrides PCIDV1 72h[3:2])</td><td colspan="2">DACK[7:0]#
Suspend state:
00 = Drive
01 = Tristate
10 = Tristate
11 = Reserved</td><td colspan="2">RTCAS, RTCRD#, RTCWR#
Suspend state:
00 = Drive
01 = Tristate
10 = RTCAS: Tristate, pull-up, RTCRD# and RTCWR#: Tristate, pull-down
11 = Reserved</td></tr>
</table>

4.15.6.2 Zero Volt CPU Suspend

The FireStar CPU interface provides a zero volt Suspend option. Setting ADh[5] = 1 enables zero volt CPU Suspend support. When set, FireStar will condition its outputs during Suspend assuming that the CPU has been powered down completely.

This feature is generally used in conjunction with a feature on the INIT pin which, in this case, is used as the general purpose CPU reset (not as a software reset). By setting SYSCFG ADh[3] = 1, the INIT signal will toggle on Resume from Suspend to reset a CPU that has been powered down.

4.15.6.3 Stopping IPC Clock When Not In Use

Setting SYSCFG 50h[4] = 0 stops the clock going to the internal integrated 82C206. Primarily this setting affects the 8254-type clock/timer/counter circuit. If the timer will not be used to maintain the system clock, substantial power savings can be achieved by disabling this clock, and turning off the OSC clock generator if possible.

4.15.7 Context Save Feature

FireStar is designed to support low-power Suspend and Zero-volt Suspend operations. Like most modern chipsets, the register set is complex enough that storing the system context before entering Zero-volt Suspend mode is a difficult procedure.

FireStar offers a context save mode feature. Context save mode changes all configuration registers so that they become shadow registers: the last value written to these registers by software becomes readable.

This feature is convenient because BIOS no longer needs to save a table of values in scarce SMM memory for registers defined as write-only or that read back a different value than the one written.

Context save mode is selected through PCIDV1 4Fh[3] as shown in Table 4-130.

Table 4-129 0V CPU Suspend and Stopping IPC Clock Register Bits

7	6	5	4	3	2	1	0
SYSCFG 50h			**PMU Control Register 4**				**Default = 00h**
			14.3MHz to 82C700: 0 = Enable 1 = Disable				
SYSCFG ADh			**Feature Control Register 3**				**Default = 00h**
		CPU power state in Suspend: 0 = Powered 1 = 0 Volt		INIT operation: 0 = Normal 1 = Toggle on Resume			

Table 4-130 Context Save Mode Programming Register Bit

7	6	5	4	3	2	1	0
PCIDV1 4Fh			**Miscellaneous Control Register C - Byte 1**				**Default = 20h**
				Context Save mode: 0 = Disable 1 = Enable			

4.15.8 Resume Event

A certain set of interrupt events can be enabled to Resume the system from the Suspend mode. The desired interrupts are grouped into a single event, called RSMGRP. RSMGRP can be enabled to generate an SMI if desired. The RINGI input and the SUS/RES input can also trigger a Resume, and can also be enabled to generate an SMI if desired. Any one or more of the RSMGRP, RINGI, and SUS/RES events are called a Resume event.

4.15.8.1 EPMI/IRQ Events

SYSCFG 6Ah and B1h select the EPMI and IRQ source(s) that will be allowed to trigger the system out of the Suspend mode. Once selected, setting SYSCFG 5Fh[5] = 1 enables the RSMGRP globally. On Resume, an SMI can be generated either from the EPMI events (through SYSCFG 58h and D9h) or PMI#6 event (through SYSCFG 59h). However, since the system usually is still in SMM when the Resume takes place, SMI generation is not normally necessary.

The default for all the IRQ and EPMI Resume enabling bits (refer to) is disabled. A rising edge on the enabled signal causes the Resume event for all selections except IRQ8; it is a falling edge on IRQ8 that Resumes operation.

Table 4-131 EPMI/IRQ Resume Event Related Register Bits

7	6	5	4	3	2	1	0
SYSCFG 58h			**PMU Event Register 1**				**Default = 00h**
LOWBAT PMI#3 SMI: 00 = Disable 11 = Enable		EPMI1# PMI#2 SMI: 00 = Disable 11 = Enable		EPMI0# PMI#1 SMI: 00 = Disable 11 = Enable		LLOWBAT PMI#0 SMI: 00 = Disable 11 = Enable	
SYSCFG 59h			**PMU Event Register 2**				**Default = 00h**
		Resume INTRGRP PMI#6, Suspend PMI#7 SMI: 00 = Disable 11 = Enable		R_TIMER PMI#5 SMI: 00 = Disable 11 = Enable		IDLE_TIMER PMI#4 SMI: 00 = Disable 11 = Enable	
SYSCFG 5Fh			**PMU Control Register 6**				**Default = 00h**
		RSMGRP IRQs can Resume system: 0 = No 1 = Yes					
SYSCFG 6Ah			**RSMGRP IRQ Register 1**				**Default = 00h**
EPMI1# Resume: 0 = Disable 1 = Enable	EPMI0# Resume: 0 = Disable 1 = Enable	IRQ8 Resume: 0 = Disable 1 = Enable	IRQ7 Resume: 0 = Disable 1 = Enable	IRQ5 Resume: 0 = Disable 1 = Enable	IRQ4 Resume: 0 = Disable 1 = Enable	IRQ3 Resume: 0 = Disable 1 = Enable	IRQ1 Resume: 0 = Disable 1 = Enable
SYSCFG 6Bh			**Resume Source Register**				**Default = 00h**
						RSMGRP caused Resume (RO): 0 = No 1 = Yes	
SYSCFG B1h			**RSMGRP IRQ Register 2**				**Default = 00h**
EPMI3# Resume: 0 = Disable 1 = Enable	EPMI2# Resume: 0 = Disable 1 = Enable	IRQ15 Resume: 0 = Disable 1 = Enable	IRQ14 Resume: 0 = Disable 1 = Enable	IRQ12 Resume: 0 = Disable 1 = Enable	IRQ11 Resume: 0 = Disable 1 = Enable	IRQ10 Resume: 0 = Disable 1 = Enable	IRQ9 Resume: 0 = Disable 1 = Enable

Table 4-131 EPMI/IRQ Resume Event Related Register Bits (cont.)

7	6	5	4	3	2	1	0
SYSCFG D9h			**PMU Event Register 6**				**Default = 00h**
DOZE_TIMER PMI#27 SMI: 00 = Disable 01 = Enable DOZE_0 10 = Enable DOZE_1 11 = Enable both		RINGI PMI#26 SMI: 00 = Disable 11 = Enable		EPMI3# cool-down clocking PMI#25 SMI: 00 = Disable 11 = Enable		EPMI2# PMI#24 SMI: 00 = Disable 11 = Enable	

4.15.8.2 SUS/RES and RINGI Events

When the RINGI input pin changes state and back enough to exceed the count set in SYSCFG 5Fh[3:0], and SYSCFG 5Fh[4] = 1, PMI#6 is generated to exit the Suspend mode. RINGI should be high for a minimum of 240ms and low for a minimum of 60ms when changing states. The RINGI input is sampled with a 32KHz clock; therefore rapid or unstable transitions may lead to unreliable counting.

The SUS/RES input pin is always enabled to Resume the system, and should be pulled high to VCC if it will not be used in the system design. Resuming from SUS/RES generates PMI#7.

Once a Resume event has occurred, SYSCFG 6Bh[2:0] should be read to determine the source(s). If SYSCFG 6Bh[1] = 1, a read of SYSCFG 6Ah and B1h will return the latched state of the any of the EPMI or IRQ lines that were originally enabled for Resume triggering. The latched Resume IRQ and EPMI source information in SYSCFG 6Ah and B1h is available until the PMI#6 bit (SYSCFG 5Ch[6]) is eventually written to 1 to clear the PMI generated. SYSCFG 50h[1] = 1 as long as the Resume PMI#6 remains active.

SYSCFG 61h[5:4] controls the debounce rate and polarity of SUS/RES. These bits function as follows:

00 Active low, edge triggered PMI. PMI#7 is triggered on any high-to-low edge of SUS/RES. Once the PMI is triggered, software must write SYSCFG 5Ch[7] = 1 to clear PMI#7 and deassert SMI#.

To Resume: Once the system is in the Suspend mode, the next high-to-low edge on SUS/RES will Resume operation.

01 Active low, level-controlled PMI. Setting SUS/RES low causes PMI#7 to go active; setting SUS/RES high causes PMI#7 to go inactive. There is no latching associated with this function, so it is not necessary to write bit 5Ch[7] = 1 to deassert the SMI#.

To Resume: A low signal on SUS/RES generates a resume function. Therefore, hardware/software must ensure that SUS/RES is high before going into Suspend mode; otherwise, the system will Resume immediately.

10 Active high, level-sampled PMI in 16ms. SUS/RES must be sampled high for at least three 4ms clock edges before being recognized as a PMI. Therefore, it takes a maximum of 16ms for the SUS/RES request to be recognized. Once the PMI is triggered, software must write bit 5Ch[7] = 1 to clear PMI#7 and deassert SMI#. Also, the SUS/RES pin must be sampled low for four clock edges (20ms maximum) before the circuit is re-armed to generate the next PMI#7.

To Resume: Once the system is in the Suspend mode, a high level sampled on SUS/RES for three 4ms clocks will resume operation.

11 Active high, level-sampled PMI in 32ms. Same as above, but the sampling clock is 8ms instead of 4ms. Therefore, SUS/RES must be sampled high for a maximum of 32ms before being recognized as a PMI, and must remain low for 40ms before the circuit is re-armed.

To Resume: Once the system is in Suspend mode, a high level sampled on SUS/RES for three 8ms clocks will resume operation.

These settings make the SUS/RES function much more practical in a design where the switch is set to one specific level to command Suspend mode, and to the other level to command Resume mode.

Example:

A notebook design incorporates a lid switch that normally leaves SUS/RES low during operation. SYSCFG 61h[5:4] are normally set to 11. When the lid is closed, SUS/RES goes high. FireStar asserts SMI# 32ms later. Software services the SMI and recognizes PMI#7 active.

The software then prepares the system for the Suspend mode and reprograms SYSCFG 61h[5:4] = 00 to prepare for Resuming. Finally, the software writes SYSCFG 50h[0] = 1 to engage the Suspend mode.

Later the lid is raised and SUS/RES goes low. Because SYSCFG 61h[5:4] = 00, the high-to-low edge on SUS/RES generates a resume and PMI#7. Software then writes SYSCFG 61h[5:4] back to 11, clears PMI#7 by writing bit 5Ch[7] = 1, and returns to normal operation.

Table 4-132 SUS/RES and RINGI Resume Event Related Register Bits

7	6	5	4	3	2	1	0
SYSCFG 50h			**PMU Control Register 4**				**Default = 00h**
					Ready to Resume (RO): 0 = Not in Resume 1 = Ready to Resume	PMU mode (RO): 0 = Nothing pending 1 = Suspend active (clear PMI#6)	
SYSCFG 5Ch			**PMI SMI Source Register 1 (Write 1 to Clear)**				**Default = 00h**
PMI#7, Suspend: 0 = Not Active 1 = Active	PMI#6, Resume or INTRGRP: 0 = Not Active 1 = Active						
SYSCFG 5Fh			**PMU Control Register 6**				**Default = 00h**
		RSMGRP IRQs can Resume system: 0 = No 1 = Yes	Transitions on RINGI can Resume system: 0 = No 1 = Yes	Number of RINGI transitions to cause Resume			
SYSCFG 61h			**Debounce Register**				**Default = 00h**
		SUS/RES# debounce rate select: 00 = Active low, edge-trig'd PMI 01 = Active low, level-controlled PMI 10 = Active high, level-sampled PMI in 16ms 11 = Active high, level-sampled PMI in 32ms (See Section 4.15.8.2, "SUS/RES and RINGI Events")					
SYSCFG 6Bh			**Resume Source Register**				**Default = 00h**
	PREQ# caused Resume (RO): 0 = No 1 = Yes	CLKRUN# caused Resume (RO): 0 = No 1 = Yes		CISA SEL#/ATB# low caused Resume (RO): 0 = No 1 = Yes	SUSP/RSM caused Resume (RO): 0 = No 1 = Yes	RSMGRP caused Resume (RO): 0 = No 1 = Yes	RI caused Resume (RO): 0 = No 1 = Yes

4.15.9 Power Control Latch

There are 16 peripheral power pins (PPWR[15:0]) that are used to control power to individual peripherals through external 74373 latches. Each latch pin is controlled with its individual control bits in SYSCFG 54h, 55h, ABh, and EEh. (Refer to Table 4-133.)

The value latched by PPWRL from the SA bus extends from PPWR15 through PPWR0, providing useful power control signals for up to 16 devices if all 16 bits are latched. Power control pins can also be derived from unused DACK# and SA pins as described below.

4.15.9.1 Power Control Pins

Certain PPWR outputs can be generated directly on the DACK[7:0]# pins on an as-needed basis. This feature eliminates the need for external TTL for a power control latch if fewer than the full number of DMA channels are employed.

When a DACK# pin is reassigned as a PPWR pin, the corresponding DRQ pin for that channel becomes available for programming as a general purpose PIO pin. The programming is done through the DMA Channel Selection registers (PCIDV1 C0h-C3h).

Certain other PPWR pins can be generated directly on SA[23:18], TC, AEN, and RFSH#. PPWR0# is available if the PPWRL signal is not used. Refer to Section 3.4, "Strap Selected Options" for more information on PPWR pin selection.

4.15.9.2 Hardware Considerations

The power control scheme uses the ISA bus SA[15:0] signals to additionally provide the inputs to a 74373-type latch. If all 16 signals will be used, two '373 devices are needed. The PPWRL signal from FireStar is active high and latches the SA[15:0] signals on the latch output on their falling edges.

The pins PPWR1 and PPWR0 have a recovery delay time associated with them when doing the Suspend/Resume function. These two pins can be used as a delay control for some component that needs some time to become stable once power is restored. For example, after turning off the power to the clock oscillator during the Suspend mode, the Resume function will restore power to the clock oscillator and wait until the clock has had time to stabilize before continuing the Resume process.

During reset, the PPWRx latch signals (PPWRL1 and PPWRL0) are pulsed to set the PPWRx signals to a known state. After reset:

- PPWR[11:8] and PPWR[3:0] are set to 0 and
- PPWR[15:12] and PPWR[7:4] are set to 1.

The PPWRx signals will remain in this state until they are updated by writing to SYSCFG 54h, 55h, ABh, and EEh.

4.15.9.3 Programming

SYSCFG 54h, 55h, ABh, and EEh set the power control latch outputs. The upper bits [7:4] of each register select whether the corresponding bits [3:0] should be used to change the latch; if the enable bit is 0, the current latch setting will not be changed when the register is written.

Table 4-133 Power Control Register Bits

7	6	5	4	3	2	1	0
SYSCFG 54h			**Power Control Latch Register 1**				**Default = 00h**
Enable [3:0] to write latch lines PPWR[3:0]: 0 = Disable 1 = Enable				Read/write data bits for PPWR[3:0]: 0 = Latch output low 1 = Latch output high			
SYSCFG 55h			**Power Control Latch Register 2**				**Default = 0Fh**
Enable [3:0] to write latch lines PPWR[7:4]: 0 = Disable 1 = Enable				Read/write data bits for PPWR[7:4] (Default = 1111): 0 = Latch output low 1 = Latch output high			
SYSCFG ABh			**Power Control Latch Register 3**				**Default = 00h**
Enable [3:0] to write latch lines PPWR[11:8]: 0 = Disable 1 = Enable				Read/write data bits for PPWR[11:8]: 0 = Latch output low 1 = Latch output high			
SYSCFG EEh			**Power Control Latch Register 4**				**Default = 0Fh**
Enable [3:0] to write latch lines PPWR[15:12]: 0 = Disable 1 = Enable				Read/write data bits for PPWR[15:12] (Default = 1111): 0 = Latch output low 1 = Latch output high			

Table 4-133 Power Control Register Bits (cont.)

7	6	5	4	3	2	1	0
PCIDV1 C0h			**DMA Channels A and B Selection Register**				**Default = 10h**
Reserved	DMA channel selection on DRQB/DACKB# pins (Channel 1): 000 = Channel 0 100 = PPWR5 001 = Channel 1 101 = Channel 5 010 = Channel 2 110 = Channel 6 011 = Channel 3 111 = Channel 7			Reserved	DMA channel selection on DRQA/DACKA# pins (Default = Channel 0): 000 = Channel 0 100 = PPWR4 001 = Channel 1 101 = Channel 5 010 = Channel 2 110 = Channel 6 011 = Channel 3 111 = Channel 7		
PCIDV1 C1h			**DMA Channels C and D Selection Register**				**Default = 32h**
Reserved	DMA channel selection on DRQD/DACKD# pins (Channel 3): 000 = Channel 0 100 = PPWR7 001 = Channel 1 101 = Channel 5 010 = Channel 2 110 = Channel 6 011 = Channel 3 111 = Channel 7			Reserved	DMA channel selection on DRQC/DACKC# pins (Channel 2): 000 = Channel 0 100 = PPWR6 001 = Channel 1 101 = Channel 5 010 = Channel 2 110 = Channel 6 011 = Channel 3 111 = Channel 7		
PCIDV1 C2h			**DMA Channel E Selection Register**				**Default = 50h**
Reserved	DMA channel selection on DRQE/DACKE# pins (Default = Channel 5): 000 = Channel 0 100 = PPWR13 001 = Channel 1 101 = Channel 5 010 = Channel 2 110 = Channel 6 011 = Channel 3 111 = Channel 7			Reserved	Function selection on TC pin: 0 = TC 1 = PPWR10	Function selection on AEN pin: 0 = AEN 1 = PPWR11	Function selection on RFSH# pin: 0 = RFSH# 1 = PPWR12
PCIDV1 C3h			**DMA Channels F and G Selection Register**				**Default = 76h**
Reserved	DMA channel selection on DRQG/DACKG# pins (Default = Channel 7): 000 = Channel 0 100 = PPWR15 001 = Channel 1 101 = Channel 5 010 = Channel 2 110 = Channel 6 011 = Channel 3 111 = Channel 7			Reserved	DMA channel selection on DRQF/DACKF# pins (Default = Channel 6): 000 = Channel 0 100 = PPWR14 001 = Channel 1 101 = Channel 5 010 = Channel 2 110 = Channel 6 011 = Channel 3 111 = Channel 7		

4.15.9.4 Resume Recovery Time

SYSCFG 68h[3:2] determine the recovery time from PPWR[1:0] active after a Resume until the end of reset. The clock is guaranteed to be active for at least the last one-eighth of the recovery time. These bits are not affected by SYSCFG 68h[1:0].

These bits can be overridden by setting SYSCFG BEh[0] = 1, in which case the Resume recovery time will always be one second.

4.15.9.5 PPWR[1:0] Suspend Auto Toggle Feature

SYSCFG 68h[1:0] enable PPWR1 and PPWR0, respectively, to automatically toggle when entering and exiting Suspend. Using PPWR0 as an example: When bit 0 = 1 and FireStar has gone into the Suspend mode, PPWR0 gets set to the inverse of SYSCFG 54h[0]; mask SYSCFG 54h[4] is ignored. When exiting Suspend, PPWR0 is set to SYSCFG 54h[0] setting, followed by the recovery time delay set in bits 68h[3:2] before continuing the Resume operation.

Table 4-134 Resume Recovery and Suspend Auto Toggle Register Bits

7	6	5	4	3	2	1	0
SYSCFG 54h			**Power Control Latch Register 1**				**Default = 00h**
Enable [3:0] to write latch lines PPWR[3:0]: 0 = Disable 1 = Enable				Read/write data bits for PPWR[3:0]: 0 = Latch output low 1 = Latch output high			
SYSCFG 68h			**Clock Source Register 3**				**Default = 00h**
				Resume recovery time: 00 = 8ms 10 = 128ms 01 = 32ms 11 = 30μs **Note:** Ignored if BEh[0] = 1.		PPWR[1:0] auto-toggle on entry and exit from Suspend: 0 = Disable 1 = Enable	
SYSCFG BEh if AEh[7] = 0			**Idle Reload Event Enable Register 2**				**Default = 00h**
							Override SYSCFG 68h[3:2]: 0 = No 1 = Recover time 1s

4.15.10 Programmable Chip Select Feature

FireStar provides programmable chip select features that require no chip signals to be sacrificed. A total of four programmable chip selects are available and can decode either memory cycles or I/O cycles. For I/O chip select decoding, granularity can be specified to-the-byte, decoding a total of 10 bits. For ROM chip select decoding, granularity is to 16KB blocks anywhere in the ISA bus address space (16MB).

Note that the memory chip select feature should be used cautiously for ROMs residing below 1MB. Since the ROM to be selected is on the SD bus, the XD bus buffer may be directed toward FireStar for memory reads and could conflict with SD bus ROMs.

The registers that control the programmability and relocation of the general purpose chip selects are shown in Table 4-135.

Table 4-135 Programmable Chip Select Registers

7	6	5	4	3	2	1	0
SYSCFG 4Ah			**Chip Select 0 Base Address Register**				**Default = 00h**
GPCS0# base address: A[8:1] (I/O) or A[22:15] (Memory)							
SYSCFG 4Bh			**Chip Select 0 Control Register**				**Default = 00h**
GPCS0# base address: A9 (I/O) A23 (Memory)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	Chip select active: 0 = w/Cmd 1 = Before ALE	GPCS0# mask bits for address A[4:1] (I/O) or A[18:15] memory: A 1 in a particular bit means that the corresponding bit at SYSCFG 4Ah[3:0] is not compared. This is used to determine address block size.			
SYSCFG 4Ch			**Chip Select 1 Base Address Register**				**Default = 00h**
GPCS1# base address: A[8:1] (I/O) or A[22:15] (Memory)							
SYSCFG 4Dh			**Chip Select 1 Control Register**				**Default = 00h**
GPCS1# base address: A9 (I/O) A23 (Memory)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	Chip select active: 0 = w/Cmd 1 = before ALE	GPCS1# mask bits for address A[4:1] (I/O) or A[18:15] memory: A 1 in a particular bit means that the corresponding bit at SYSCFG 4Ch[3:0] is not compared. This is used to determine address block size.			
SYSCFG B3h			**Chip Select Cycle Type Register**				**Default = 00h**
GPCS3# ROM width: 0 = 8-bit 1 = 16-bit	GPCS2# ROM width: 0 = 8-bit 1 = 16-bit	GPCS1# ROM width: 0 = 8-bit 1 = 16-bit	GPCS0# ROM width: 0 = 8-bit 1 = 16-bit	GPCS3# cycle type: 0 = I/O 1 = ROMCS	GPCS2# cycle type: 0 = I/O 1 = ROMCS	GPCS1# cycle type: 0 = I/O 1 = ROMCS	GPCS0# cycle type: 0 = I/O 1 = ROMCS
SYSCFG BAh			**Chip Select 2 Base Address Register**				**Default = 00h**
GPCS2# base address: A[8:1] (I/O) or A[22:15] (Memory)							
SYSCFG BBh			**Chip Select 2 Control Register**				**Default = 00h**
GPCS2# base address: A9 (I/O) A23 (Memory)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	Chip select active: 0 = w/Cmd 1 = before ALE	GPCS2# mask bits for address A[4:1] (I/O) or A[18:15] memory: A 1 in a particular bit means that the corresponding bit at SYSCFG BAh[3:0] is not compared. This is used to determine address block size.			
SYSCFG BCh			**Chip Select 3 Base Address Register**				**Default = 00h**
GPCS3# base address: A[8:1] (I/O) or A[22:15] (Memory)							

Table 4-135 Programmable Chip Select Registers (cont.)

7	6	5	4	3	2	1	0
SYSCFG BDh			**Chip Select 3 Control Register**				**Default = 00h**
GPCS3# base address: A9 (I/O) A23 (Memory)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	Chip select active: 0 = w/Cmd 1 = before ALE	GPCS3# mask bits for address A[4:1] (I/O) or A[18:15] memory: A 1 in a particular bit means that the corresponding bit at SYSCFG BCh[3:0] is not compared. This is used to determine address block size.			
SYSCFG BFh			**Chip Select Granularity Register**				**Default = 0Fh**
GPCS3# base address: A0 (I/O) A14 (Memory)	GPCS2# base address: A0 (I/O) A14 (Memory)	GPCS1# base address: A0 (I/O) A14 (Memory)	GPCS0# base address: A0 (I/O) A14 (Mem.)	GPCS3# mask bit: A0 (I/O) A14 (Memory)	GPCS2# mask bit: A0 (I/O) A14 (Memory)	GPCS1# mask bit: A0 (I/O) A14 (Memory)	GPCS0# mask bit: A0 (I/O) A14 (Memory)

4.15.10.1 XDIR Control on GPCS# Ranges

FireStar provides a register (see Table 4-136) to control whether GPCS# decoding also controls the XD bus buffer direction. Since FireStar implements the XD bus buffer internally, using GPCS# decoding is the only way to control buffer direction for non-standard X bus devices.

The same control register also provides bits to individually enable and disable the decode. This feature somewhat duplicates the bits already in the individual GPCS# programming registers that enable decoding separately for reads and writes on each GPCS# line. However, changing those bits requires multiple reads and writes; the global bits are provided here to allow a simplified means of temporarily disabling the decode. These bits default to "enabled" to remain backward compatible with the Viper-N+ BIOS.

Table 4-136 GPCS# Control Bits

SYSCFG FEh			**GPCS# Global Control Register**				**Default = 00h**
GPCS3# decode: 0 = Enable 1 = Disable	GPCS2# decode: 0 = Enable 1 = Disable	GPCS1# decode: 0 = Enable 1 = Disable	GPCS0# decode: 0 = Enable 1 = Disable	GPCS3# read cycles drive XDIR low: 0 = Disable 1 = Enable	GPCS2# read cycles drive XDIR low: 0 = Disable 1 = Enable	GPCS1# read cycles drive XDIR low: 0 = Disable 1 = Enable	GPCS0# read cycles drive XDIR low: 0 = Disable 1 = Enable

4.16 ACPI Implementation

This section of the data book describes how the Microsoft Advanced Configuration and Power Interface (ACPI) is implemented in the FireStar silicon. ACPI is a standard register interface for power management functions jointly developed by Microsoft, Intel, and Toshiba. The features required by ACPI closely mirror the feature set already implemented in FireStar. In many cases, support of ACPI entails only a remapping of register bits already present in FireStar.

The ACPI register set does not replace the standard FireStar PMU register set, but is instead available in parallel. When ACPI is enabled, both register sets can be used interchangeably. Therefore, existing power management software can continue to run unchanged until the ACPI code is fully capable of replacing it.

4.16.1 Register Level Interface

ACPI defines two Power Management register blocks PM1_BLK and PM2_BLK, and Processor register block P_BLK, to accommodate most on-chip power management operations. ACPI further defines the optional General Purpose Event register blocks GPE0_BLK and GPE1_BLK to accommodate external events that require power management intervention. FireStar implements only GPE0_BLK so that GPE1_BLK can be implemented by another PCI device in the system if needed.

Many of the functions addressed by these bits correspond to existing FireStar features. The correspondence is noted where appropriate.

Table 4-137 is an overview of the above mentioned register blocks (the bit meanings are explained later in this chapter.)

Table 4-137 Register Block Overview

Offset	7	6	5	4	3	2	1	0
PM1_BLK Register Set								
00h	Reserved		GBL_STS	BM_STS	Reserved			TMR_STS
01h	WAK_STS	Reserved			PWRBTNOR_STS	RTC_STS	Reserved	PWRBTN_STS
02h	Reserved		GBL_EN	Reserved				TMR_EN
03h	Reserved					RTC_EN	Reserved	PWRBTN_EN
04h	Reserved					GBL_RLS	BM_RLD	SCI_EN
05h	Reserved		SLP_EN	SLP_TYP			Reserved	
06h-07h	Reserved							
08h	TMR_VAL[7:0]							
09h	TMR_VAL[15:8]							
0Ah	TMR_VAL[23:16]							
0Bh-0Dh	Reserved							
PM2_BLK Register Set								
00h	Reserved							ARB_DIS
01h-07h	Reserved							
P_BLK Register Set								
00h	Reserved			THT_EN	CLK_VAL			Reserved
01h-03h	Reserved							
04h	P_LVL2							
05h	P_LVL3							
06h-07h	Reserved							
GPE0_BLK Register Set (bit positions not specified by ACPI)								
00h	ACPI7 LID_STS	ACPI6 EC_STS	ACPI5 USB_STS	ACPI4 RI_STS	ACPI3 FRI_STS	ACPI2 STSCHG_STS	ACPI1 DOCK_STS	ACPI0 UNDOCK_STS
01h	Reserved			THRM_STS	ACPI11_STS	ACPI10_STS	ACPI9_STS	ACPI8_STS
02h	ACPI7 LID_EN	ACPI6 EC_EN	ACPI5 USB_EN	ACPI4 RI_EN	ACPI3 FRI_EN	ACPI2 STSCHG_EN	ACPI1 DOCK_EN	ACPI0 UNDOCK_EN
03h	BIOS_RLS	Reserved		THRM_EN	ACPI11_EN	ACPI10_EN	ACPI9_EN	ACPI8_EN
04h-07h	Reserved							

4.16.2 Base Address Selection

The Base Address selection method for PM1_BLK, PM2_BLK, P_BLK, and GPE0_BLK is chipset-specific. For FireStar, these base addresses are individually selectable at any 16-bit I/O address (on dword boundaries only). These locations are programmed by the ACPI BIOS after a call from the OS, using the FireStar PCIDV1 register set, as shown in Table 4-138.

In addition to the Base Address selection registers, Table 4-138 shows other ACPI related registers located in the PCIDV1 register space. These registers' features are individually addressed and discussed in the following sub-sections.

Table 4-138 PCIDV1 ACPI Related Registers

7	6	5	4	3	2	1	0
PCIDV1 D0h		**PM1_BLK Base Address Register - Byte 0: Address Bits [7:0]**					**Default = 00h**
PM1 Block Base Address Bits - Address value A[15:0] defines the 16-bit starting address for PM1_BLK Register Set in system I/O space. The address is required to be paragraph-aligned (on a 16-byte boundary), so bits [3:0] are always 0.							PM1_BLK Register Set: 0 = Disable 1 = Enable
PCIDV1 D1h		**PM1_BLK Base Address Register - Byte 1: Address Bits [15:8]**					**Default = 00h**
PCIDV1 D2h		**PM2_BLK Base Address Register - Byte 0: Address Bits [7:0]**					**Default = 00h**
PM2 Block Base Address Bits - Address value A[15:0] defines the 16-bit starting address for PM2_BLK Register Set in system I/O space. The address is required to be qword-aligned (on an 8-byte boundary), so bits [2:0] are always 0.							PM2_BLK Register Set: 0 = Disable 1 = Enable
PCIDV1 D3h		**PM2_BLK Base Address Register -Byte 1: Address Bits [15:8]**					**Default = 00h**
PCIDV1 D4h		**P_BLK Base Address Register - Byte 0: Address Bits [7:0]**					**Default = 00h**
Processor Block Base Address Bits - Address value A[15:0] defines the 16-bit starting address for P_BLK Register Set in system I/O space. The address is required to be qword-aligned (on an 8-byte boundary), so bits [2:0] are always 0.							P_BLK Register Set: 0 = Disable 1 = Enable
PCIDV1 D5h		**P_BLK Base Address Register - Byte 1: Address Bits [15:8]**					**Default = 00h**
PCIDV1 D6h		**GPE0_BLK Base Address Register - Byte 0: Address Bits [7:0]**					**Default = 00h**
General Purpose Event Block Base Address Bits - Address value A[15:0] defines the 16-bit starting address for GPE0_BLK Register Set in system I/O space. The address is required to be qword-aligned (on an 8-byte boundary), so bits [2:0] are always 0.							GPE0_BLK Register Set: 0 = Disable 1 = Enable
PCIDV1 D7h		**GPE0_BLK Base Address Register - Byte 0: Address Bits [15:8]**					**Default = 00h**
PCIDV1 D8h			**ACPI Source Control Register - Byte 0**				**Default = 00h**
ACPI7 LID: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI6 EC#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI5 USB#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI4 RI#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI3 FRI#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI2 STSCHG#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI1 DOCK#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI0 UNDOCK#: 0 = IRQ Driveback 1 = Discrete ACPI input

Table 4-138 PCIDV1 ACPI Related Registers (cont.)

7	6	5	4	3	2	1	0
PCIDV1 D9h			**ACPI Source Control Register - Byte 1**				**Default = 00h**
Reserved				ACPI11: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI10: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI9: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI8: 0 = IRQ Driveback 1 = Discrete ACPI input
Note: The bits in the ACPI Source Control Register (Bytes 0 and 1) select whether the specified ACPI input comes from the IRQ Driveback cycle or from an external pin source (one of the PIO pins or through ACPIMX option).							
PCIDV1 DAh			**ACPI Source Status Register - Byte 0**				**Default = 00h**
ACPI7 LID: 0 = Low 1 = High	ACPI6 EC#: 0 = Low 1 = High	ACPI5 USB#: 0 = Low 1 = High	ACPI4 RI#: 0 = Low 1 = High	ACPI3 FRI#: 0 = Low 1 = High	ACPI2 STSCHG#: 0 = Low 1 = High	ACPI1 DOCK#: 0 = Low 1 = High	ACPI0 UNDOCK#: 0 = Low 1 = High
PCIDV1 DBh			**ACPI Source Status Register - Byte 1**				**Default = 00h**
Reserved				ACPI11: 0 = Low 1 = High	ACPI10: 0 = Low 1 = High	ACPI9: 0 = Low 1 = High	ACPI8: 0 = Low 1 = High
Note: The bits in the ACPI Source Status Register (Bytes 0 and 1) indicate the current state of the ACPI lines, either the discrete pins or the last IRQ Driveback value depending on the ACPI Source Control Register setting. This information may also be available elsewhere, since the IRQ Driveback values and PIO pin values can be read from other registers. However, this register provides a central means of reading signal state and is especially useful for signals such as LID (which generates an SCI, System Controller Interrupt, on both opening and closing events).							
PCIDV1 DCh			**ACPI Event Resume Control Register - Byte 0**				**Default = 00h**
ACPI7 LID: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI6 EC#: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI5 USB#: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI4 RI#: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI3 FRI#: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI2 STSCHG#: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI1 DOCK#: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI0 UNDOCK#: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend
PCIDV1 DDh			**ACPI Event Resume Control Register - Byte 1**				**Default = 00h**
Reserved				ACPI11: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI10: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI9: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI8: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend
Note: The bits in the ACPI Event Resume Control Register (Bytes 0 and 1) select whether the specified ACPI input can wake the system from its Suspend mode. Note that any PCI device that sends its information via the IRQ Driveback cycle will wake the system when it activates its CLKRUN# pin.							

Table 4-138 PCIDV1 ACPI Related Registers (cont.)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="3">PCIDV1 DEh-DFh</td><td colspan="3">Reserved</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="3">PCIDV1 E0h</td><td colspan="3">SLP_TYP Control Register - Byte 0</td><td colspan="2">Default = 00h</td></tr>
<tr><td>PLVL17:
Selects state PPWR17 line will assume when SCTL_ PPWR17 setting is reached.
0 = Low
1 = High</td><td colspan="3">SCTL_PPWR17:
Refer to bits [2:0] for decode.</td><td>PLVL16:
Selects state PPWR16 line will assume when SCTL_ PPWR16 setting is reached.
0 = Low
1 = High</td><td colspan="3">SCTL_PPWR16:
000 = PPWRx switches when SLP_TYP (PM1_BLK Offset 05h[4:2) is set for ACPI S0 system state
001 = PPWRx switches when SLP_TYP (PM1_BLK Offset 05h[4:2) is set for ACPI S0 or S1 system state
010 = PPWRx switches when SLP_TYP (PM1_BLK Offset 05h[4:2) is set for ACPI S0, S1, or S2 system state
011 = PPWRx switches when SLP_TYP (PM1_BLK Offset 05h[4:2) is set for ACPI S0, S1, S2, or S3 system state
100 = PPWRx switches when SLP_TYP (PM1_BLK Offset 05h[4:2) is set for ACPI S0, S1, S2, S3, or S4 system state</td></tr>
<tr><td colspan="3">PCIDV1 E1h</td><td colspan="3">SLP_TYP Control Register - Byte 1</td><td colspan="2">Default = 00h</td></tr>
<tr><td>PLVL19:
Selects state PPWR19 line will assume when SCTL_ PPWR19 setting is reached.
0 = Low
1 = High</td><td colspan="3">SCTL_PPWR19:
Refer to PCIDV1 E0h[2:0] for decode.</td><td>PLVL18:
Selects state PPWR18 line will assume when SCTL_ PPWR18 setting is reached.
0 = Low
1 = High</td><td colspan="3">SCTL_PPWR18:
Refer to PCIDV1 E0h[2:0] for decode.</td></tr>
<tr><td colspan="3">PCIDV1 E2h</td><td colspan="3">SLP_TYP Control Register - Byte 2</td><td colspan="2">Default = 00h</td></tr>
<tr><td>PLVL21:
Selects state PPWR21 line will assume when SCTL_ PPWR21 setting is reached.
0 = Low
1 = High</td><td colspan="3">SCTL_PPWR21:
Refer to PCIDV1 E0h[2:0] for decode.</td><td>PLVL20:
Selects state PPWR20 line will assume when SCTL_ PPWR20 setting is reached.
0 = Low
1 = High</td><td colspan="3">SCTL_PPWR20:
Refer to PCIDV1 E0h[2:0] for decode.</td></tr>
<tr><td colspan="3">PCIDV1 E3h</td><td colspan="3">SLP_TYP Control Register - Byte 3</td><td colspan="2">Default = 00h</td></tr>
<tr><td>PLVL23:
Selects state PPWR23 line will assume when SCTL_ PPWR23 setting is reached.
0 = Low
1 = High</td><td colspan="3">SCTL_PPWR23:
Refer to PCIDV1 E0h[2:0] for decode.</td><td>PLVL22:
Selects state PPWR22 line will assume when SCTL_ PPWR22 setting is reached.
0 = Low
1 = High</td><td colspan="3">SCTL_PPWR22:
Refer to PCIDV1 E0h[2:0] for decode.</td></tr>
</table>

Table 4-138 PCIDV1 ACPI Related Registers (cont.)

7	6	5	4	3	2	1	0
PCIDV1 E4h			**SLP_TYP Control Register - Byte 4**				**Default = 00h**
PLVL25: Selects state PPWR25 line will assume when SCTL_ PPWR25 setting is reached. 0 = Low 1 = High	SCTL_PPWR25: Refer to PCIDV1 E0h[2:0] for decode.			PLVL24: Selects state PPWR24 line will assume when SCTL_ PPWR24 setting is reached. 0 = Low 1 = High	SCTL_PPWR24: Refer to PCIDV1 E0h[2:0] for decode.		
PCIDV1 E5h			**SLP_TYP Control Register - Byte 5**				**Default = 00h**
PLVL27: Selects state PPWR27 line will assume when SCTL_ PPWR27 setting is reached. 0 = Low 1 = High	SCTL_PPWR27: Refer to PCIDV1 E0h[2:0] for decode.			PLVL26: Selects state PPWR26 line will assume when SCTL_ PPWR26 setting is reached. 0 = Low 1 = High	SCTL_PPWR26: Refer to PCIDV1 E0h[2:0] for decode.		
PCIDV1 E6h			**SLP_TYP Control Register - Byte 6**				**Default = 00h**
PLVL29: Selects state PPWR29 line will assume when SCTL_ PPWR29 setting is reached. 0 = Low 1 = High	SCTL_PPWR29: Refer to PCIDV1 E0h[2:0] for decode.			PLVL28: Selects state PPWR28 line will assume when SCTL_ PPWR28 setting is reached. 0 = Low 1 = High	SCTL_PPWR28: Refer to PCIDV1 E0h[2:0] for decode.		
PCIDV1 E7h			**SLP_TYP Control Register - Byte 7**				**Default = 00h**
PLVL31: Selects state PPWR31 line will assume when SCTL_ PPWR31 setting is reached. 0 = Low 1 = High	SCTL_PPWR31: Refer to PCIDV1 E0h[2:0] for decode.			PLVL30: Selects state PPWR30 line will assume when SCTL_ PPWR30 setting is reached. 0 = Low 1 = High	SCTL_PPWR30: Refer to PCIDV1 E0h[2:0] for decode.		
PCIDV1 E8h			**Power Control Latch Set Register**				**Default = 00h**
Control line setting: 0 = Low 1 = High	Reserved		PPWR control line to be set: 00000 = PPWR0 00001 = PPWR1 ... ... 11110 = PPWR30 11111 = PPWR31				
PCIDV1 E9h			**Reserved**				**Default = 00h**

Table 4-138 PCIDV1 ACPI Related Registers (cont.)

7	6	5	4	3	2	1	0
PCIDV1 EAh			**Power Control Readback Register - Byte 0**				**Default = FFh**
PPWR7 state: 0 = Low 1 = High	PPWR6 state: 0 = Low 1 = High	PPWR5 state: 0 = Low 1 = High	PPWR4 state: 0 = Low 1 = High	PPWR3 state: 0 = Low 1 = High	PPWR2 state: 0 = Low 1 = High	PPWR1 state: 0 = Low 1 = High	PPWR0 state: 0 = Low 1 = High
PCIDV1 EBh			**Power Control Readback Register - Byte 1**				**Default = FFh**
PPWR15 state: 0 = Low 1 = High	PPWR14 state: 0 = Low 1 = High	PPWR13 state: 0 = Low 1 = High	PPWR12 state: 0 = Low 1 = High	PPWR11 state: 0 = Low 1 = High	PPWR10 state: 0 = Low 1 = High	PPWR9 state: 0 = Low 1 = High	PPWR8 state: 0 = Low 1 = High
PCIDV1 ECh			**Power Control Readback Register - Byte 2**				**Default = F0h**
PPWR23 state: 0 = Low 1 = High	PPWR22 state: 0 = Low 1 = High	PPWR21 state: 0 = Low 1 = High	PPWR20 state: 0 = Low 1 = High	PPWR19 state: 0 = Low 1 = High	PPWR18 state: 0 = Low 1 = High	PPWR17 state: 0 = Low 1 = High	PPWR16 state: 0 = Low 1 = High
PCIDV1 EDh			**Power Control Readback Register - Byte 3**				**Default = F0h**
PPWR31 state: 0 = Low 1 = High	PPWR30 state: 0 = Low 1 = High	PPWR29 state: 0 = Low 1 = High	PPWR28 state: 0 = Low 1 = High	PPWR27 state: 0 = Low 1 = High	PPWR26 state: 0 = Low 1 = High	PPWR25 state: 0 = Low 1 = High	PPWR24 state: 0 = Low 1 = High
PCIDV1 EEh			**ACPI Thermal Control Register**				**Default = 00h**
Reserved		PIO pin FAN control is auto-toggled high: 00 = Never 01 = During Level 1 and 2 STPCLK# modulation 10 = During Level 2 STPCLK# modulation only 11 = Reserved		Temperature event granularity: Selects the bit of the THFREQ value in SYSCFG F3h-F4h that will be monitored such that it generates a thermal management event when it toggles. 0000 = Bit 0, 0001 = Bit 1, 0010 = Bit 2, 0011 = Bit 3 0100 = Bit 4, 0101 = Bit 5, 0110 = Bit 6, 0111 = Bit 7 1000 = Bit 8, 1001 = Bit 9, 1010 = Bit 10, 1011 = Bit 11 1100 = Bit 12, 1101 = Bit 13, 1110 = Bit 14, 1111 = Bit 15			

4.16.2.1 PM1 Register Block

The PM1 registers themselves are always located in system I/O space as offsets from the PM1_BLK I/O Base Address, PCIDV1 D0h-D1h.

A register map for the PM1 register set is shown in Table 4-139 with relevant bit information following.

Table 4-139 PM1_BLK Register Set

Offset	7	6	5	4	3	2	1	0
00h	Reserved		GBL_STS	BM_STS	Reserved			TMR_STS
01h	WAK_STS	Reserved			PWRBTNOR_STS	RTC_STS	Reserved	PWRBTN_STS
02h	Reserved		GBL_EN	Reserved				TMR_EN
03h	Reserved					RTC_EN	Reserved	PWRBTN_EN
04h	Reserved					GBL_RLS	BM_RLD	SCI_EN
05h	Reserved		SLP_EN	SLP_TYP			Reserved	
06h-07h	Reserved							
08h	TMR_VAL[7:0] (RO)							
09h	TMR_VAL[15:8] (RO)							
0Ah	TMR_VAL[23:16] (RO)							
0Bh-0Dh	Reserved							

SCI Enable

ACPI takes over power management by setting SCI_EN = 1. Until this point, any enabled event will generate an SMI (PMI#39, indicated active in SYSCFG DDh[7]).

- SCI_EN
  - If SCI occurs, generate:
    0 = SMI
    1 = IRQ13

24-bit Timer

The timer is a free-running "up" counter based on the 14MHz clock divided by 4. It runs whenever the 14MHz input clock to FireStar is present, and is cleared to 0 whenever PCIRST# is asserted. The TMR_VAL register is read-only. Whenever TMR_VAL[23] changes from 0-to-1 or from 1-to-0, the TMR_STS bit is set to 1; writing 1 back to TMR_STS clears the bit. If TMR_EN = 1 when TMR_STS = 1, an SCI occurs (if globally enabled).

- TMR_STS
  - Has TMR_VAL[23] toggled (changed from high-to-low or low-to-high)?
    0 = No
    1 = Yes
    Write 1 to clear
- TMR_EN
  - Should TMR_STS going to 1 cause SCI?
    0 = No
    1 = Yes
- TMR_VAL[23:0]
  - A read-only value that returns the power management timer count. The count is based on 14.31818MHz/4. The count is cleared by a PCI bus reset. Whenever bit 23 toggles, TMR_STS is set to indicate the event. Counts only while the system is active.

Bus Master Monitor

BM_STS goes high whenever a PCI REQ# line goes active (or an internal bus master device such as the IDE controller makes a bus request). Software writes back a 1 to clear the bit. No SCI can be generated by this event. However, if BM_RLD = 1, BM_STS going active returns the CPU to C0 (full active) state from C3 state.

- BM_STS
  - Has any REQ# gone active since this bit was last cleared?
    0 = No
    1 = Yes
    Write 1 to clear
- BM_RLD
  - Should BM_STS going to 1 wake up CPU (state restored to C0 from C3)?
    0 = No
    1 = Yes

Global Service Request

BIOS can get the attention of ACPI, or ACPI can get the attention of BIOS, through the Global Service Request bits.

BIOS request to ACPI:
BIOS writes 1 to BIOS_RLS (offset from GPE0_BLK 03h[7]), which will cause GBL_STS to be set to 1 until cleared (when ACPI code writes GBL_STS = 1). If GBL_EN = 1, GBL_STS going to 1 causes an SCI. BIOS_RLS is write-only; reads always return 0.

- BIOS_RLS
  - Does BIOS want to make a request to ACPI?
    0 = No effect
    1 = Yes
    Write-only bit
- GBL_STS
  - Has software written BIOS_RLS = 1?
    0 = No
    1 = Yes
    Write 1 to clear
- GBL_EN
  - Should GBL_STS going to 1 cause SCI?
    0 = No
    1 = Yes

ACPI request to BIOS:
ACPI writes 1 to GBL_RLS, which will cause a software SMI. This bit is connected directly to SYSCFG 50h[7]. When BIOS services the SMI, it must clear the software SMI (by writing either GBL_RLS or SYSCFG 50h[7]).

- GBL_RLS
  - Does ACPI software wish to generate SMI to BIOS?
    0 = No
    1 = Yes

Wakeup Status

Uses bit WAK_STS to indicate that the system was in Suspend and was woken up due to an enabled Resume event. Similar to FireStar SYSCFG 5Ch[6].

- WAK_STS
  - Did system wake from Suspend mode after an enabled Resume event occurred?
    0 = No
    1 = Yes

Power Button

The PWRBTN# signal operation is exactly the same as the FireStar SUSP/RES pin operation when SYSCFG 61h[5:4] = 00. Driving PWRBTN# low when the system is active causes PWRBTN_STS to be set to 1, and will also cause an SCI if PWRBTN_EN = 1.

PWRBTN# also has an additional “override” function. If PWRBTNOR_EN = 1 and the PWRBTN# signal is held active for more than four seconds, the system is forced to the “off” state with no software involved. In this case, PWRBTN_STS gets cleared to 0; PWRBTNOR_STS gets set to 1.

On FireStar, the override feature is implemented by forcing a write of SYSCFG 50h[0] = 1 after four seconds, which will do a Suspend sequence and toggle the PPWR pins.

- PWRBTN_STS
  - Has user pressed power button?
    0 = No
    1 = Yes
- PWRBTN_EN
  - Should PWRBTN_STS going to 1 cause SCI?
    0 = No
    1= Yes
- PWRBTNOR_STS
  - PWRBTN# Override Status - PWRBTN# asserted for > 4 sec?
    0 = No
    1 = Yes

RTC Management

The RTC interrupt, IRQ8#, can be used to generate an SCI event. Whenever IRQ8# goes active, RTC_STS goes to 1. If RTC_EN = 1 also, an SCI is generated.

- RTC_STS
  - Has IRQ8# from RTC gone active?
    0 = No
    1 = Yes
- RTC_EN
  - Should RTC_STS going to 1 cause SCI?
    0 = No
    1 = Yes

Sleep Modes

ACPI software writes SLP_TYP to any desired value along with SLP_EN = 1 to force entry into a Sleep mode. SCTL_PPWRx bits (in the chipset-specific registers described earlier) select how the PPWR control lines will react to each SLP_TYP selection.

- SLP_EN
  - When written to 1, forces SLP_TYP Suspend mode. Always reads 0.
- SLP_TYP
  - Defines Sleep mode to enter when software sets SLP_EN = 1. ACPI ROM table associates 3-bit binary values with one of the system states S0-S4.

- SCTL_PPWRx
  - The SCTL_PPWRx (Sleep control) bits select the SLP_TYP modes for which the associated PPWR (peripheral device control line output from FireStar) should be controlled. If SCTL_PPWRx <= SLP_TYP, the PPWRx line will be controlled. "Controlled" means set to the PLVLx value associated with that PPWRx line.

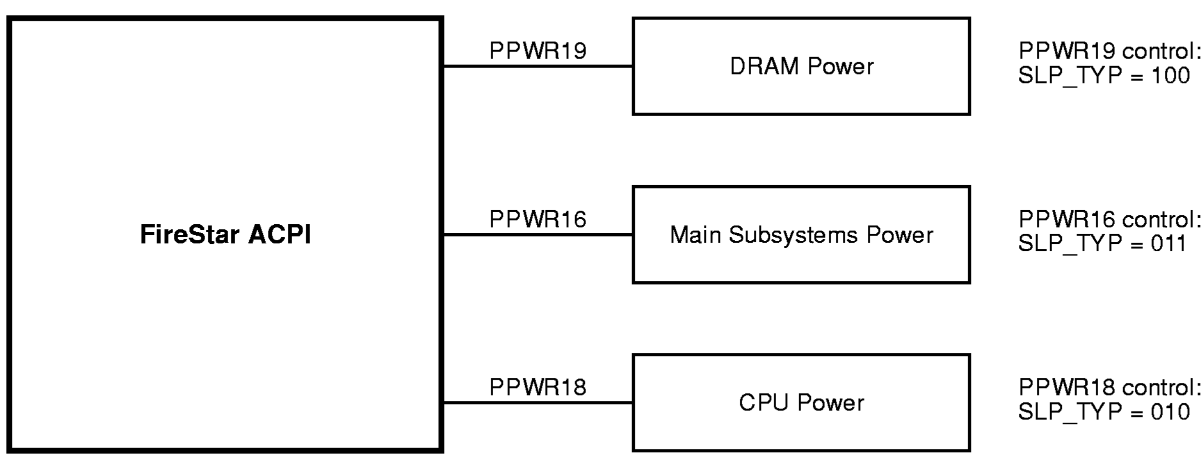
    For example, if SCTL_PPWR9 = 010, PLVL9 = 0, and the current state of PPWR9 is high, PPWR9 will be switched to low on entry to SLP_TYP mode 2 and higher but not when SLP_TYP = 1. On exit from Sleep, PPWR9 will be driven to its previous value.

ACPI defines five system states S0-S4.

- S0, SLP_TYP = 000
  - Active mode. Clock throttling, etc. determined by CPU state C0-C3.
- S1, SLP_TYP = 001
  - Low-power Suspend mode with CPU and L2 cache alive. Selecting SLP_TYP = 011 causes SYSCFG ADh[5,3] to be set to 0,0 to allow powered CPU Suspend mode, and then forces a write to SYSCFG 50h[0] to start the transition into Suspend mode. WAK_STS = 1 upon resume.
- S2, SLP_TYP = 010
  - Same as S1, but power is removed from CPU, L2 cache, and selected peripheral devices. Selecting SLP_TYP = 010 causes SYSCFG ADh[5,3] to be set to 1,1 to allow 0V CPU Suspend mode, and then forces a write to SYSCFG 50h[0] to start the transition into Suspend mode. Auto-toggle occurs on selected PPWR pins to allow power to be removed from the devices automatically in the Suspend process. WAK_STS = 1 upon resume.
- S3, SLP_TYP = 011
  - Same as S2, but power is removed from more devices. Typically, only DRAM, FireStar, and the keyboard controller (embedded controller) remain powered. Auto-toggle on PPWR lines is enabled more broadly to distinguish between S2 and S3.
- S4 - SLP_TYP = 100
  - Same as S3, but power is also removed from DRAM in this mode. Auto-toggle on PPWR lines is enabled more broadly to distinguish between S3 and S4.

A typical system organization is illustrated in Figure 4-41.

Figure 4-41 FireStar SLP_TYP Power Control Scheme Example



4.16.2.2 PM2 Register Block

The registers addresses shown in Table 4-140 are offsets from the PM2_BLK I/O Base Address, PCIDV1 D2h-D3h.

Table 4-140 PM2_BLK Register Set

Offset	7	6	5	4	3	2	1	0
00h	Reserved							ARB_DIS
01h-07h	Reserved							

ARB_DIS

- Software uses this bit to enable and disable system master devices. When set to 1, only the host CPU can own the system buses. When cleared to 0, other requesting bus masters will be granted the bus in the normal manner. This feature allows software to initiate time-critical operations without the possibility of another device disrupting the operation.
  0 = Enable arbitration
  1 = Disable

4.16.2.3 Processor Register Block

The registers addresses in Table 4-141 are offsets from the P_BLK I/O Base Address, PCIDV1 D4h-D5h.

Table 4-141 P_BLK Register Set

Offset	7	6	5	4	3	2	1	0
00h	Reserved			THT_EN	CLK_VAL			Reserved
01h-03h	Reserved							
04h	P_LVL2							
05h	P_LVL3							
06h-07h	Reserved							

Clock Throttling

THT_EN and CLK_VAL bits work in conjunction with each other. The CLK_VAL bits correspond to FireStar SYSCFG 67h[2:0]. However, since the duty cycle mapping is not exact, the FireStar duty cycle for each range will be adjusted to the upper ACPI limit (001 = 12.5% ... 111 = 87.5%).

- THT_EN
  - Throttle enable - Enables clock throttling.
    0 = Disable
    1 = Enable
- CLK_VAL
  - Clock throttle duty - Sets STPCLK# throttling duty cycle.
    000 = Reserved
    001 = 0-12.5%
    010 = 12.5-25%
    011 = 25-37.5%
    100 = 37.5%-50%
    101 = 50-62.5%
    110 = 62.5-75%
    111 = 75%-87.5%

CPU States

ACPI defines four CPU states C0-C3. The P_BLK registers control entry to levels C2 and C3.

- P_LVL2
  - Force Power Level 2 - Reading this register forces clock control logic to C2 state. Writes are ignored.
    (SYSCFG 50h[3] - APM Doze Mode)
- P_LVL3
  - Force Power Level 3 - Reading this register forces clock control logic to C3 state. Writes are ignored.
    (SYSCFG 50h[0] - 0V CPU Suspend Mode)

Hardware events control exit from any level to level C0:

- C0
  - CPU working state - Full speed operation on FireStar.
- C1
  - CPU low power state 1 - Achieved by software generating HLT instruction to CPU.
- C2
  - CPU low power state 2 - Achieved by software reading P_LVL2 register, which causes chipset to assert STPCLK# to the CPU. Same as FireStar APM Doze Mode.
- C3
  - CPU low power state 3 - Achieved by software reading P_LVL3 register, which causes chipset to assert STPCLK# to the CPU and then stop the clocks to the CPU. Same as FireStar APM Doze Mode, with the L2CLKOE signal wired to control CPU clocks as well as L2 cache clocks.

The original FireStar PMU hardware already implements most of the logic necessary to perform these operations. What is necessary in addition is a way to distinguish between P_LVL2 and P_LVL3. FireStar does this using the L2CLKOE signal.

4.16.2.4 General Purpose Bits

The General Purpose register bits are offsets from the GPE0_BLK I/O Base Address, PCIDV1 D6h-D7h. The register bit positions for these blocks are not specifically defined by Microsoft. The OPTi mapping is shown in Table 4-142.

Table 4-142 GPE0_BLK Register Set

Offset	7	6	5	4	3	2	1	0
00h	ACPI7 LID_STS	ACPI6 EC_STS	ACPI5 USB_STS	ACPI4 RI_STS	ACPI3 FRI_STS	ACPI2 STSCHG_STS	ACPI1 DOCK__STS	ACPI0 UNDOCK_STS
01h	Reserved			THRM_STS	ACPI11_STS	ACPI10_STS	ACPI9_STS	ACPI8_STS
02h	ACPI7 LID_EN	ACPI6 EC_EN	ACPI5 USB_EN	ACPI4 RI_EN	ACPI3 FRI_EN	ACPI2 STSCHG_EN	ACPI1 DOCK_EN	ACPI0 UNDOCK_EN
03h	BIOS_RLS	Reserved		THRM_EN	ACPI11_EN	ACPI10_EN	ACPI9_EN	ACPI8_EN
04h-07h	Reserved							

On OPTi Mobile products, certain status information (for example, USB_STS and RI_STS) arrives through the IRQ Driveback cycle. The mapping of IRQ Driveback bits to STS change events is described in Section 4.16.3, FireStar IRQ Driveback Feature.

Embedded Event

This event occurs whenever the EC# signal goes low, indicating an interrupt from the embedded controller (keyboard controller).

- EC_STS
  - Embedded Controller Event Status - Set if EC# line goes low.
- EC_EN
  - Used to enable SCI from EC_STS event:
    0 = Disable
    1 = Enable

USB Event

This event occurs whenever the USB# signal goes low from the USB controller.

- USB_STS
  - USB# Signal Status - Set if USB# line goes low.
- USB_EN
  - Used to enable SCI from USB_STS event:
    0 = Disable
    1 = Enable

Ring Indicator Events

The RI event occurs whenever the RI# signal goes low. It is assumed that RI# will arrive from another device (Super I/O chip, etc.) that will provide the telephone line filtering.

- RI_STS
  - RI# Signal Status - Set if RI# goes low (local pin or from IRQ driveback).
- RI_EN
  - Used to enable SCI from RI_STS event:
    0 = Disable
    1 = Enable

The FRI event occurs whenever the FRI# pin senses an input frequency between 12Hz and 68Hz, the range set aside by telephone companies for ring signalling. Modem cards put out a 5.0V signal that toggles with this frequency.

A frequency detection is used that only counts rising edges (since the duty cycle is undefined). The reason for the frequency detection circuit is that the phone lines are very noisy, which makes detecting a single transition unreliable and causes false wakeup events.

- FRI_STS
  - FRI# Signal Status - Set if FRI# goes low (local pin or from IRQ driveback).
- FRI_EN
  - Used to enable SCI from FRI_STS event:
    0 = Disable
    1 = Enable

Thermal Event

This event occurs when the monitored bit of the THFREQ value specified in TEMPGR[3:0] (PCIDV1 EEh[3:0]) toggles.

- THRM_STS
  - Has the bit of the THFREQ value specified in TEMPGR[3:0] toggled?
    0 = No
    1 = Yes
- THRM_EN
  - Enable SCI on thermal event:
    0 = No
    1 = Yes

Lid Status Change Event

This event occurs whenever the LID signal goes high or low. Note that this signal is different from others in that it generates an event on both high- and low-going edges.

- LID_STS
  - Lid open or close event status.
- LID_EN
  - Enable SCI on lid open/close event:
    0 = No
    1 = Yes

PCMCIA Status Change Event

This event occurs whenever the STSCHG# signal goes low if provided from a PCMCIA controller.

- STSCHG_STS
  - Status change event status.
- STSCHG_EN
  - Enable SCI on change event:
    0 = No
    1 = Yes

Dock/Undock Event

This event occurs whenever the DOCK# signal goes low if provided from a docking controller.

- DOCK_STS
  - Docking or undocking event status.
- DOCK_EN
  - Enable SCI on dock or undock event:
    0 = No
    1 = Yes

Undock Request Event

This event occurs whenever the UNDOCK# signal goes low. UNDOCK# is usually provided from a switch that is either local or on the docking station.

- UNDOCK_STS
  - Undock request event status.
- UNDOCK_EN
  - Enable SCI on undock request:
    0 = No
    1 = Yes

BIOS Request

BIOS can get the attention of ACPI, or ACPI can get the attention of BIOS, through the Global Service Request bits. This bit, BIOS_RLS, works in conjunction with the Global Service Request bits (PM1_BLK Register Set). See "Global Service Request" on page 224 for further information.

- BIOS_RLS
  - Does BIOS want to make a request to ACPI?
    0 = No effect
    1 = Yes
  - Write-only bit; reads always return 0.

4.16.3 FireStar IRQ Driveback Feature

FireStar depends on the IRQ Driveback feature to obtain external event information from other OPTi Mobile products without using any pins. The IRQ Driveback cycle occurs when an external event takes place. The interrupting device becomes master of the PCI bus, and then writes its updated status information to a preprogrammed port address in the FireStar I/O space.

Two dwords are typically returned during an IRQ Driveback cycle. As seen in the tables below, Phase 1 returns IRQ information; Phase 2 returns PCI PCIRQ# information and ACPI event information. The mapping of ACPI0-11 to ACPI events such as DOCK_STS and USB_STS is provided in Section 4.16.5, "ACPI-to-EPMI Mapping, Muxing".

Note that the chipset always handles the ACPI bits as edge-generating events into edge-triggered logic. For example, if one device generates a cycle with ACPI7 = 1 (assuming ENACPI7# = 0), the event sets the corresponding General Purpose Register status bit if enabled. Software will write the status bit to 1 to clear it. If another device generates another cycle with ACPI7 = 1 before the first device performs an IRQ driveback with ACPI7 = 0, it will still cause the GP register status bit to be set to 1. The cycle in which ACPI7 is restored to 0 has no effect.

Table 4-143 Information provided on Data Phase 1 of IRQ Driveback Cycle

Low Word	AD15	AD14	AD13	AD12	AD11	AD10	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
	IRQ15	IRQ14	IRQ13 (NMI)	IRQ12	IRQ11	IRQ10	IRQ9	IRQ8	IRQ7	IRQ6	IRQ5	IRQ4	IRQ3	IRQ2 (SMI)	IRQ1	IRQ0
High Word	AD31	AD30	AD29	AD28	AD27	AD26	AD25	AD24	AD23	AD22	AD21	AD20	AD19	AD18	AD17	AD16
	EN15#	EN14#	EN13#	EN12#	EN11#	EN10#	EN9#	EN8#	EN7#	EN6#	EN5#	EN4#	EN3#	EN2#	EN1#	EN0#

Table 4-144 Information provided on Data Phase 2 of IRQ Driveback Cycle

Low Word	AD15	AD14	AD13	AD12	AD11	AD10	AD9	AD8	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0
	ACPI 11	ACPI 10	ACPI 9	ACPI 8	ACPI 7	ACPI 6	ACPI 5	ACPI 4	ACPI 3	ACPI 2	ACPI 1	ACPI 0	PCIRQ 3	PCIRQ 2	PCIRQ 1	PCIRQ 0
High Word	AD31	AD30	AD29	AD28	AD27	AD26	AD25	AD24	AD23	AD22	AD21	AD20	AD19	AD18	AD17	AD16
	EN ACPI 11#	EN ACPI 10#	EN ACPI 9#	EN ACPI 8#	EN ACPI 7#	EN ACPI 6#	EN ACPI 5#	EN ACPI 4#	EN ACPI 3#	EN ACPI 2#	EN ACPI 1#	EN ACPI 0#	EN P3#	EN P2#	EN P1#	EN P0#

4.16.4 Power Control

The ACPI version of FireStar will provide a total of 32 power control pins, as opposed to the 16 power control pins that are available on the initial FireStar revisions. Because the requirement for the ISA bus is expected to disappear during the lifetime of the FireStar class chips, the ACPI power control latch is now based on the SD[15:0] bus instead of the SA bus. The SD bus will always be available on FireStar class chips as a general purpose data port.

The PPWRL pin now takes on the additional role of PCTLH (Power Control Latch - High). On initial production FireStar chips with the standard PMU, PPWRL latches SA[15:0] to become PPWR[15:0]. On ACPI-enabled FireStar chips, PPWR[31:16] also become available and are latchable from SD[15:0].

PCTLL (Power Control Latch - Low) is a new PIO pin option available only on the RSTDRV pin. PCTLL is used to latch PPWR[15:0] from SD[15:0]. If ISA devices require the RSTDRV signal, it should be supplied by inverting the RESET# output signal from FireStar.

In summary:

- PPWR[31:16] are latched from SD[15:0] using PCTLH (existing PPWRL) pin. These pins can be auto-toggled.
- PPWR[15:0] are latched from SD[15:0] using PCTLL (RSTDRV) pin. These pins cannot be auto-toggled.
- PPWR[15:0] are available from SA[15:0] using PCTLH (existing PPWRL) pin for backward compatibility. In this case, PPWR[1:0] can be auto-toggled.
- PWR[31:16] are only available through the external latch. However, PPWR[15:0] are available as dedicated pins on spare PIO pins. PPWR[1:0] can be auto-toggled when supplied from PIO pins.

Note that PPWR[1:0] will auto-toggle when recovered from the SA bus bits [1:0], or when output on discrete I/O pins, but not when recovered from the SD bus using PCTLL. In new designs it is recommended to recover PPWR pins only from the SD bus, and to assign auto-toggle functions only to pins PPWR[31:16].

4.16.4.1 PPWR Control Register

The PPWR lines have a control mechanism under FireStar ACPI: PCI configuration registers are used to set the control lines and to read back their state. The registers are shown in Table 4-145.

BIOS code should use only these registers to set the PPWR lines, not the SYSCFG registers. No SYSCFG registers are provided to access PPWR[31:16].

Table 4-145 PPWR Control Registers

7	6	5	4	3	2	1	0
PCIDV1 E8h			**Power Control Latch Set Register**				**Default = 00h**
Control line setting: 0 = Low 1 = High	Reserved		PPWR control line to be set: 00000 = PPWR0 00001 = PPWR1 ... ... 11110 = PPWR30 11111 = PPWR31				
PCIDV1 E9h			**Reserved**				**Default = 00h**
PCIDV1 EAh			**Power Control Readback Register - Byte 0**				**Default = FFh**
PPWR7 state: 0 = Low 1 = High	PPWR6 state: 0 = Low 1 = High	PPWR5 state: 0 = Low 1 = High	PPWR4 state: 0 = Low 1 = High	PPWR3 state: 0 = Low 1 = High	PPWR2 state: 0 = Low 1 = High	PPWR1 state: 0 = Low 1 = High	PPWR0 state: 0 = Low 1 = High
PCIDV1 EBh			**Power Control Readback Register - Byte 1**				**Default = FFh**
PPWR15 state: 0 = Low 1 = High	PPWR14 state: 0 = Low 1 = High	PPWR13 state: 0 = Low 1 = High	PPWR12 state: 0 = Low 1 = High	PPWR11 state: 0 = Low 1 = High	PPWR10 state: 0 = Low 1 = High	PPWR9 state: 0 = Low 1 = High	PPWR8 state: 0 = Low 1 = High
PCIDV1 ECh			**Power Control Readback Register - Byte 2**				**Default = F0h**
PPWR23 state: 0 = Low 1 = High	PPWR22 state: 0 = Low 1 = High	PPWR21 state: 0 = Low 1 = High	PPWR20 state: 0 = Low 1 = High	PPWR19 state: 0 = Low 1 = High	PPWR18 state: 0 = Low 1 = High	PPWR17 state: 0 = Low 1 = High	PPWR16 state: 0 = Low 1 = High
PCIDV1 EDh			**Power Control Readback Register - Byte 3**				**Default = F0h**
PPWR31 state: 0 = Low 1 = High	PPWR30 state: 0 = Low 1 = High	PPWR29 state: 0 = Low 1 = High	PPWR28 state: 0 = Low 1 = High	PPWR27 state: 0 = Low 1 = High	PPWR26 state: 0 = Low 1 = High	PPWR25 state: 0 = Low 1 = High	PPWR24 state: 0 = Low 1 = High

The recommended mapping of PPWR lines to power control functions is shown in the tables that follow. PPWR[31:16], which are capable of auto-toggle (switching with only the 32KHz clock stable) are assigned to the signals that require this function.

Table 4-146 PPWR Power Control Defaults

Power Control Defaults Latched from SA Bus															
PPWR 15	PPWR 14	PPWR 13	PPWR 12	PPWR 11	PPWR 10	PPWR 9	PPWR 8	PPWR 7	PPWR 6	PPWR 5	PPWR 4	PPWR 3	PPWR 2	PPWR 1	PPWR 0
High	High	High	High	Low	Low	Low	Low	High	High	High	High	Low	Low	Low	Low
Power Control Defaults Latched from SD Bus															
PPWR 15	PPWR 14	PPWR 13	PPWR 12	PPWR 11	PPWR 10	PPWR 9	PPWR 8	PPWR 7	PPWR 6	PPWR 5	PPWR 4	PPWR 3	PPWR 2	PPWR 1	PPWR 0
High	High	High	High	High	High	High	High	High	High	High	High	High	High	High	High
PPWR 31	PPWR 30	PPWR 28	PPWR 28	PPWR 27	PPWR 26	PPWR 25	PPWR 24	PPWR 23	PPWR 22	PPWR 21	PPWR 20	PPWR 19	PPWR 18	PPWR 17	PPWR 16
High	High	High	High	Low	Low	Low	Low	High	High	High	High	Low	Low	Low	Low

Table 4-147 Recommended PPWR Control Assignments

Area of Control	ACPI Signal	FireStar Pin	Auto-toggle Function	Default State	Comment
Main System	CLK_EN	PPWR20	Enabled	High	Main clock generator control
	RAMCLK_EN#	PPWR17	Enabled	Low	Control for clocks going to sync DRAM, L2 cache
	CPU_PWR#	PPWR18	Powered	Low	CPU power control
	DRAM_PWR#	PPWR19	Powered	Low	DRAM power control
Graphics Subsystem	GR_PWR#	PPWR24	Powered	Low	
	GR_RST#	PPWR25	Reset	Low	
	GR_SUS#	PPWR5		High*	
	GR_CLKPWR#	PPWR26	Powered	Low	
	GR_CLKEN#	PPWR27	Enabled	Low	
Audio Subsystem	AUD_PWR#	PPWR6		High*	
	AUD_RST	PPWR7		High*	
	AUD_ISO	PPWR28	Isolated	High	
	ADU_SUS#	PPWR12		High	
	AMP_SUS#	PPWR13		High	
IDE 0	IDE0_PWR#	PPWR16	Powered	Low	Only main drive is powered at reset
	IDE0_RST#	PPWR8		High*	
	IDE0_ISO	DBE0#	Isolated	--	Isolation occurs automatically when DBE0# is used
IDE 1	IDE1_PWR#	PPWR21	Off	High	
	IDE1_RST#	PPWR9		High*	
	IDE1_ISO	DBE1#	Isolated	--	Isolation occurs automatically when DBE1# is used
IDE 2	IDE2_PWR#	PPWR22	Off	High	
	IDE2_RST#	PPWR10		High*	
	IDE2_ISO	DBE2#	Isolated	--	Isolation occurs automatically when DBE2# is used
IDE 3	IDE3_PWR#	PPWR23	Off	High	
	IDE3_RST#	PPWR11		High*	
	IDE3_ISO	DBE3#	Isolated	--	Isolation occurs automatically when DBE3# is used
Floppy Disk Controller	FDD_PWR#	PPWR14		High*	
	FDD_ISO	PPWR15		High*	
Communication Ports	PRT0_RST#	PPWR2		High*	
	PRT0_ISO	PPWR29	Isolated	High	
	PRT0_SUS#	PPWR0		High*	
	PRT1_RST#	PPWR3		High*	
	PRT1_ISO	PPWR30	Isolated	High	
	PRT1_SUS#	PPWR1		High*	
Miscellaneous	CB_SUS#	PPWR31		High	Suspend control for CardBus controllers not designed with automatic power management
		PPWR4		High*	Unassigned

*Assuming signals are latched from SD bus.

- Assumptions made:
  - Isolation is enabled when control signal is high (buffer enable inputs are usually active when low).
  - Devices powered on when control line is low.
  - CPU clock generator is enabled when control signal is high.

IDE Control
The ACPI specification recommends a power switch scheme for independent control of IDE drives 0-3. In the signal descriptions below, 'x' refers to one of drives 0-3.

- IDEx_PWR# - Enables power to the drive when active (low). A PPWRx pin is used to implement this function.
- IDEx_RST# - Resets the drive when active (low). A PPWRx pin is used to implement this function.
- IDEx_ISO# - Isolates the drive when active (low). OPTi Mobile logic automatically tristates drive buffers between cycles by deasserting the DBE# signal associated with that drive. Note that the signal sense is opposite to that of the ACPI-specified signal.

Audio Control
The ACPI specification recommends a power switch scheme for independent control of the main and amplifier sections of the audio subsystem.

- AUD_PWR# - Enables power to the main audio chip when active (low). A PPWRx pin is used to implement this function.
- AUD_RST - Resets the (ISA) audio chip when active (high). A PPWRx pin is used to implement this function.
- AUD_ISO# - Isolates the audio interface from the rest of the system when active (low). A PPWRx pin is used to implement this function.
- AUD_SUS# - Puts the powered main audio chip into its lowest power standby mode when active (low). A PPWRx pin is used to implement this function.
- AMP_SUS# - Puts the audio amplifier into its lowest power standby mode when active (low), possibly by removing power to the amp. A PPWRx pin is used to implement this function.

Graphics Control
The ACPI specification recommends the following control scheme for the graphics subsystem.

- GR_PWR# - Enables power to the central graphics logic when active (low). There is no GR_ISO# signal, as it is assumed that the graphics chip has the isolation function built in. A PPWRx pin is used to implement the GR_PWR# function.
- GR_RST# - Resets the graphics chip when active (low). A PPWRx pin is used to implement this function.
- GR_SUS# - Puts the powered graphics chip into its lowest power standby mode when active (low). A PPWRx pin is used to implement this function.
- GR_CLKPWR# - Enables power to the graphics clock generator circuit when active (low). A PPWRx pin is used to implement this function.
- GR_CLKEN# - Enables clocks to the graphics chip when active (low) once the generator output is stable. A PPWRx pin is used to implement this function.

Miscellaneous Control
ACPI defines control signals for several other subsystems such as the serial ports, parallel ports, and floppy disk controller. It is assumed that the Super I/O chip taking care of these functions will provide appropriate control lines, but assignments are provided here just in case they are needed.

4.16.5 ACPI-to-EPMI Mapping, Muxing

ACPI calls for several power management inputs as requirements, and suggests several others as options. In addition, customers have made requests for special inputs. The following list provides the complete set of options to-date.

- RI# - Ring Indication from modem
- FRI# - Filtered Ring Indication that signals SCI only if certain frequency is detected
- USB# - Wakeup indication from USB port
- EC# - Embedded Controller interrupt
- DOCK# - Indication that docking/undocking event occurred
- UNDOCK# - Indication that undock request has been made
- STSCHG# - Indication that status change event has occurred
- LID - Indication that lid to notebook has been opened

OPTi Mobile ACPI-capable chips allow these signals to be passed transparently between OPTi peripheral devices and the OPTi host chipset by means of the ACPI0-11 bits of the IRQ Driveback cycle, which uses no pins on either device. It is necessary only to map the ACPI0-11 bits to their corresponding ACPI signal function and polarity.

Since not all devices in a system will conform to the OPTi standard, the ACPI inputs can also be brought into the host chipset through discrete pins. There are two options for doing this.

If there is a sufficient number of spare PIO pins available on the host chipset, they can be assigned as discrete ACPI inputs through the normal PIO programming scheme. Function group 7 has been added to the PIO programming matrix to accommodate ACPI0-11.

If many ACPI inputs are needed, they can be muxed in on the ACPIMX0-2 pins (also PIO options). External 4-to-1 muxes are used to select the pin being read. This method is identical to the EPMIMUX scheme used on some OPTi chipsets, and relies on the signals ATCLK and ATCLK/2 (another PIO pin).

Table 4-148 shows how ACPI0-11 are recommended to be mapped to the ACPI signal function set. Reserved spots are for future assignment, but can be used as needed.

The mapping of ACPI event notification from either the IRQ driveback cycle or external pins is controlled by the register shown in Table 4-149.

Table 4-148 Recommended ACPI0-11 Mapping

ACPI Signal Function	Rsvd	Rsvd	Rsvd	Rsvd	LID	EC#	USB#	RI#	FRI#	STSCHG#	DOCK#	UNDOCK#
IRQ Driveback Name	ACPI 11	ACPI 10	ACPI 9	ACPI 8	ACPI 7	ACPI 6	ACPI 5	ACPI 4	ACPI 3	ACPI 2	ACPI 1	ACPI 0

Table 4-149 ACPI Source Control Register

7	6	5	4	3	2	1	0
PCIDV1 D8h			**ACPI Source Control Register - Byte 0**				**Default = 00h**
The bits in this register select whether the specified ACPI input comes from the IRQ Driveback cycle or from an external pin source (one of the PIO pins or through ACPIMX option).							
ACPI7 LID: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI6 EC#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI5 USB#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI4 RI#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI3 FRI#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI2 STSCHG#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI1 DOCK#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI0 UNDOCK#: 0 = IRQ Driveback 1 = Discrete ACPI input
PCIDV1 D9h			**ACPI Source Control Register - Byte 1**				**Default = 00h**
Reserved				ACPI11: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI10: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI9: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI8: 0 = IRQ Driveback 1 = Discrete ACPI input

4.16.6 Temperature Trip Points

FireStar incorporates temperature monitoring hardware that allows it to select trip points for temperature monitoring. ACPI has special requirements that an SCI be generated on passing periodic temperature levels, while hardware clock throttling be enforced at specific levels.

The existing FireStar temperature monitoring circuit uses a 16-bit counter to keep track of temperature. The current value of this counter is readable at SYSCFG F4h-F3h as THFREQ[15:0]. The new ACPI registers shown in Table 4-150 allow toggling on any bit in this register to generate a thermal management event.

Example

Assume that the thermal sensor hardware in a specific design causes THFREQ reflected in SYSCFG F4h-F3h to change by 1 for every 0.25°C change in CPU temperature. So a change of 4°C in CPU temperature would result in a THFREQ change of 16 (10 hex). Therefore, to generate an SCI on every 4°C change in CPU temperature, the Temperature Event Granularity setting TEMPGR[3:0] would need to be set to 3 so that every time bit 3 of THFREQ changes, an SCI would result.

Note that an SCI is generated when the selected bit toggles as the temperature is rising, but no SCI is generated when passing through that point immediately afterward; the logic uses the next most significant bit to determine this situation. For example, if TEMPGR[3:0] = Ch, pointing to bit 12, and the counter value first passes from

0101 1111 1111 1111
to
0110 0000 0000 0000

a thermal management SCI event would be generated. However, if the temperature immediately started to decrease so that bit 12 toggled back to 1 again, a new SCI would not be generated. The value would have to decrease to

0100 1111 1111 1111
or increase to
0111 0000 0000 0000

before the next SCI would occur. This operation ensures that a system that is maintaining a steady temperature will not cause excessive SCIs if the temperature is floating around a specific trip point.

4.16.6.1 Fan Control

The ACPI-enabled FireStar chips allow reaching of the LOFREQ (SYSCFG A7h-A6h) or HIFREQ (SYSCFG A9h-A8h) trip points to toggle the FAN pin (PIO option). This control line can be used to control a fan to help cool the CPU. PCIDV1 EEh[5:4] control this feature.

Table 4-150 ACPI Thermal Control Register

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">PCIDV1 EEh</td><td colspan="4">ACPI Thermal Control Register</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2">Reserved</td><td colspan="2">PIO pin FAN control is auto-toggled high:
00 = Never
01 = During Level 1 and 2 STPCLK# modulation
10 = During Level 2 STPCLK# modulation only
11 = Reserved</td><td colspan="4">Temperature event granularity:
Selects the bit of the THFREQ value in SYSCFG F3h-F4h that will be monitored such that it generates a thermal management event when it toggles.
0000 = Bit 0   0100 = Bit 4   1000 = Bit 8   1100 = Bit 12
0001 = Bit 1   0101 = Bit 5   1001 = Bit 9   1101 = Bit 13
0010 = Bit 2   0110 = Bit 6   1010 = Bit 10   1110 = Bit 14
0011 = Bit 3   0111 = Bit 7   1011 = Bit 11   1111 = Bit 15</td></tr>
</table>

4.16.7 New PIO Pin Options

ACPI-enabled FireStar-class products provide the following PIO pin options.

- PCTLL
  - Power Control Latch-Low is used to latch PPWR[15:0] from SD[15:0].
- PWRBTN#
  - Power Button is the ACPI-defined Suspend/Resume input that has a special hardware override feature to allow a forced Suspend sequence.
- ACPI0-11
  - Discrete ACPI event inputs pins are provided for devices that cannot use the OPTi IRQ Driveback cycle to convey this information.

The FireStar ACPI PIO pin options are shown in Table 4-151 and are italicized. For a complete listing of all PIO pin options refer to Table 3-3 "PIO Functions" on page 33.

Table 4-151 ACPI PIO Pin Options

Group	Function	Sub-function Number	Description
Misc. Outputs Group 3h	GPCSx#	0-3h	General Purpose Chip Select outputs, x=0-3
	Reserved	4-7h	
	CDIR	8h	Compact ISA Cable Buffer Direction signal
	L2CLKOE	9h	L2 Cache Clock Output Enable
	PCICLKOE	Ah	PCI Clock Output Enable (to ext. clock generator)
	HGNT#	Bh	UMA Split Buffer Control signal
	FAN	*Ch*	*CPU overtemp fan control output*
	Reserved	Dh	
	PCTLL	*Eh*	*Power control latch low is available only on PIO15 (RSTDRV) and is used to latch PPWR[15:0] from SD[15:0]*
	ATCLK/2	*Fh*	*ATCLK divided by 2 (for KBCLK and/or ACPIMX)*
IDE Controller Outputs Group 4h	DDACK0#	0h	Dedicated IDE DMA acknowledge (Primary cable)
	DDACK1#	1h	Dedicated IDE DMA acknowledge (Secondary cable)
	DRD#	2h	Dedicated IDE command
	DWR#	3h	
	DCS1#	4h	Dedicated IDE chip select
	DCS3#	5h	
	DA0	6h	Dedicated IDE address
	DA1	7h	
	DA2	8h	
	DBEX#	9h	IDE buffer control for drive X
	DBEY#	Ah	IDE buffer control for drive Y
	DBEZ#	Bh	IDE buffer control for drive Z
	DDACK0-0#	*Ch*	*Dedicated IDE DMA acknowledge (Primary Cable, Drive 0)*
	DDACK0-1#	*Dh*	*Dedicated IDE DMA acknowledge (Primary Cable, Drive 1)*
	DDACK1-0#	*Eh*	*Dedicated IDE DMA acknowledge (Secondary Cable, Drive 0)*
	DDACK1-1#	*Fh*	*Dedicated IDE DMA acknowledge (Secondary Cable, Drive 1)*

Table 4-151 ACPI PIO Pin Options (cont.)

Group	Function	Sub-function Number	Description
ACPI Inputs Group 7h	*UNDOCK# (ACPI0)*	*0h*	*Active low input*
	DOCK# (ACPI1)	*1h*	*Active low input*
	STSCH# (ACPI2)	*2h*	*Active low input*
	FRI# (ACPI3)	*3h*	*Active low input*
	RI# (ACPI4)	*4h*	*Active low input*
	USB# (ACPI5)	*5h*	*Active low input*
	EC# (ACPI6)	*6h*	*Active low input*
	LID (ACPI7)	*7h*	*Active high input*
	(ACPI8)	*8h*	*Active high input*
	(ACPI9)	*9h*	*Active high input*
	(ACPI10)	*Ah*	*Active high input*
	(ACPI11)	*Bh*	*Active high input*
	ACPIMX0	*Ch*	*Time-multiplexed input of ACPI0-3*
	ACPIMX1	*Dh*	*Time-multiplexed input of ACPI4-7*
	ACPIMX2	*Eh*	*Time-multiplexed input of ACPI8-11*
	PWRBTN#	*Fh*	*Power button with hardware-enforced Suspend feature*

4.17 System Management Interrupt (SMI)

The 3.3V Pentium processor offers a System Management Interrupt (SMI) that allows external logic to signal to the CPU that a high priority event has occurred and must be serviced but should not in any way interfere with the application currently being processed. When the CPU senses its SMI# input active, it saves the context of its current application and loads the context of its System Management Mode (SMM) handler routine from a protected part of RAM. SMM code can then proceed to determine the reason for the interrupt, service it appropriately, and return to application processing through a special RESUME instruction that restores the context as it originally was before the SMI.

FireStar handles 39 Power Management Interrupt (PMI) events that can be selectively enabled to cause an SMI to the CPU. Since some of these PMI events are actually a single indication from a group of events (such as a single PMI #6 that indicates whether any of the selected IRQ lines has gone active), the effective number of events that can be indicated is actually much greater than 39.

The PMI events that can be programmed to generate an SMI are listed in Table 4-152.

Table 4-152 SMI Sources

Source	PMI Name	Description
IRQ, DRQ, and EPMI SMI Sources		
#3	LOWBAT	Activity on Low Battery Pin
#0	LLOWBAT	Activity on Very Low Battery Pin
#1	EPMI0#	Activity on External Power Management Input 1
#2	EPMI1#	Activity on External Power Management Input 2
#24	EPMI2#	Activity on External Power Management Input 3
#25	EPMI3#	Activity on External Power Management Input 4
#26	RINGI	Activity detected on RINGI
#7	SUS/RES	SUS/RES input has been toggled
#6	INTRGRP - or - RSMGRP	An interrupt from the INTRGRP set has occurred while the system was running -or- An interrupt from the RSMGRP has occurred and resumed the system from Suspend mode
#28	DMA TRAP	Activity on DMA DRQ lines
#33	DOZE RELOAD	Exit from hardware Doze mode
Time-Out Event SMI Sources		
#35	APM EXIT	Exit from APM (software) Doze mode
#4	IDLE_TIMER	IDLE_TIMER has timed out due to no I/O activity
#27	DOZE_TIMER	DOZE_TIMER has timed out due to inactivity
#5	R_TIMER	R_TIMER has timed out on its normal periodic basis
#8	LCD_TIMER	LCD_TIMER has timed out because of no screen activity
#9	DSK_TIMER	Floppy (and/or external hard) disk timer has timed out because of no activity
#19	HDU_TIMER	Time-out has occurred because no access has occurred in the internal IDE range
#10	KBD_TIMER	Keyboard timer has timed out because of no controller accesses
#11	GNR1_TIMER	Time-out has occurred because the memory or I/O range selected by GNR1 has had no activity
#16	GNR2_TIMER	Time-out has occurred because the memory or I/O range selected by GNR2 has had no activity
#30	GNR3_TIMER	Time-out has occurred because the memory or I/O range selected by GNR3 has had no activity
#32	GNR4_TIMER	Time-out has occurred because the memory or I/O range selected by GNR4 has had no activity
#17	COM1_TIMER	Time-out has occurred because no access has occurred in the COM1 range
#18	COM2_TIMER	Time-out has occurred because no access has occurred in the COM2 range
Access Event SMI Sources		
#14	KBD_ACCESS	Keyboard controller has been accessed, either before or after timer time-out depending on Current/Next Access setting

Table 4-152 SMI Sources (cont.)

Source	PMI Name	Description
#12	LCD_ACCESS	LCD controller has been accessed, either before or after timer time-out depending on Current/Next Access setting
#13	DSK_ACCESS	Floppy (or external hard) disk controller has been accessed, either before or after timer time-out depending on Current/Next Access setting
#23	HDU_ACCESS	Internal IDE has been accessed, either before or after timer time-out depending on Current/Next Access setting
#15	GNR1_ACCESS	GNR1 range has been accessed, either before or after timer time-out depending on Current/Next Access setting
#20	GNR2_ACCESS	GNR2 range has been accessed, either before or after timer time-out depending on Current/Next Access setting
#29	GNR3_ACCESS	GNR3 range has been accessed, either before or after timer time-out depending on Current/Next Access setting
#31	GNR4_ACCESS	GNR4 range has been accessed, either before or after timer time-out depending on Current/Next Access setting
#21	COM1_ACCESS	COM1 has been accessed, either before or after timer time-out depending on Current/Next Access setting
#22	COM2_ACCESS	COM2 has been accessed, either before or after timer time-out depending on Current/Next Access setting
Miscellaneous Event SMI Sources		
#34	Hot Dock_TIMER	Time-out has occurred in an attempt at hot docking.
#36	Serial IRQ	An SMI has been signalled by a device using the serial interrupt line.
#37	DMA_ACCESS	A DMA controller register is being accessed
#38	IRQ_DRIVEBACK	An SMI has been signalled by a device using IRQ driveback.

4.17.1 SMI Operation and Initialization

The 3.3V Pentium CPU uses the SMIACT# pin to indicate that it is currently executing SMM code. While in SMM, the default addresses put out by the CPU are in the 3000h and 4000h segments to execute SMM code and to access SMM data. However, the SMBASE Register of the CPU is programmable, and can be set to any other segment if desired. These SMRAM addresses put out by the CPU must be mapped to the A000h-B000h segments. FireStar directs these accesses to translation is performed only during SMM when FireStar receives the SMIACT# signal from the CPU. The A000h-B000h segments of DRAM main memory are usually unused and not accessed during normal mode because accesses to this area are redirected to the ISA or local bus for video. These segments are utilized by initializing them with SMM code/data at boot-up and write protecting them during normal mode of operation.

4.17.1.1 Loading Initial SMM Code and Data

On system initialization, the system management code and data segments must be loaded from ROM with the appropriate information. This information will reside in the DRAM segments at physical starting addresses A0000h and B0000h and, once loaded, will be write-protected except when the system is operating in SMM.

Step 1: System Initialization (not in SMM)

On system initialization, the BIOS must load initial code and data into the protected SMM memory space. Normally the system will still be executing out of ROM at this point, but the memory subsystem is configured and enabled. A mechanism is provided by which the A000h-B000h DRAM area may be accessed even if the CPU is not in SMM. This mechanism is used to initialize the A000h-B000h DRAM area with SMM handler code/data.

The registers that pertain to initialization are shown in Table 4-153. SYSCFG 13h[3] is the SMRAM access control and provides a global control for address translation. The value of SYSCFG 14h[3] has different meanings according to whether the CPU is in SMM or not. If SYSCFG 13h[3] = 1, and SYSCFG 14h[3] = 1, normal mode CPU accesses in the A000h-B000h range are redirected to the DRAM A000h-B000h area. This feature is used for initializing the DRAM A000h-B000h range.

Table 4-153 SMRAM Access Control Bits

7	6	5	4	3	2	1	0
SYSCFG 13h			**Memory Decode Control Register 1**				**Default = 00h**
				SMRAM: 0 = Disable 1 = Enable See SYSCFG 14h[3]			
SYSCFG 14h			**Memory Decode Control Register 2**				**Default = 00h**
				SMRAM control: Inactive SMIACT#: 0 = Disable SMRAM 1 = Enable SMRAM(1) Active SMIACT#: 0 = Enable SMRAM for both Code and Data(1) 1 = Enable SMRAM for Code only(1)			
(1) If SYSCFG 13h[3] is set.							

SYSCFG 13h[3] and 14h[3] are used as follows when the CPU is not in SMM:

- 13h[3] = 0:
  - No relocation. This setting prevents application software from accessing SMI memory space.
- 13h[3] = 1:
  - If 14h[3] = 1, CPU addresses in the A000h-B000h segments go to SMI memory space - the DRAM segments at A000h-B000h. This setting provides the mechanism for initially loading SMI code to the A000h-B000h region.
  - If 14h[3] = 0, the A000h-B000h area in DRAM cannot be accessed.

The significance of 14h[3] during SMM is explained in Section 4.17.1.2, "Run-Time SMI Address Relocation".

The BIOS sets 13h[3] = 1 and 14h[3] = 1. It can then load code and data into DRAM segments A000h and B000h. This first load operation **must** be addressed to the A000h and B000h segments. Upon completing the loading of all initial SMM code and data, the BIOS clears 14h[3] to 0 to protect the SMM space.

Step 2: Loading the Code to Change the SMBASE Register of the CPU

Having loaded the code and data, the BIOS must now generate an SMI to enter SMM so that it can complete the SMM initialization process (changing SMBASE to A0000h and for performing system-specific tasks; the SMBASE Register can only be changed from within SMM mode). The SMBASE Register of the CPU is by default 30000h, and the first code fetch in SMM is from 38000h. Before generating an SMI, the ROM BIOS must load code from physical address 38000h onwards to change the SMBASE Register of the CPU and to resume normal mode.

Step 3: Software generation of SMI

To allow software SMI generation to take place, SYSCFG 59h[7] must be written to 1. Writing SYSCFG 50h[7] = 1 asserts SMI# to the CPU to start SMM operation. Writing SYSCFG 50h[7] = 0 clears the SMI. The SMI routine **must** clear this bit; otherwise, SMI requests will be generated continuously. (Refer to Table 4-154.)

Step 4: Reprogramming SMBASE

Once the system has entered SMM for the first time at 38000h, the CPU SMBASE value can be reprogrammed for future use.

1. SMM initialization code updates the SMBASE value in the CPU register save area to A0000h.
2. SMM initialization code clears the Software Start SMI (SYSCFG 50h[7]).
3. SMM initialization code generates a RESUME instruction to return control to the BIOS initialization code. The new SMBASE value gets written to the CPU registers.

4.17.1.2 Run-Time SMI Address Relocation

The Dynamic SMI Relocation feature provides full memory access control while in SMM. SMI relocation at run time is controlled by SYSCFG 14h[3] if SYSCFG 13h[3] = 1. (Refer back to Table 4-153 for bit definitions.)

If SYSCFG 13h[3] is set, during SMM, either all CPU accesses to the A000h-B000h range, or only accesses to code may be mapped to the A000h-B000h range in DRAM memory. The active SMIACT# signal and the status of SYSCFG 14h[3] determine whether both code and data accesses or only code accesses are mapped to DRAM. If SYSCFG 14h[3] = 1, only code accesses are mapped to DRAM and data accesses are not translated to SMI space. This allows data in the A000h-B000h memory space to be accessed and saved to disk. If SYSCFG 14h[3] = 0, both code and data accesses are translated to SMI space.

Table 4-154 Software SMI Enable Register Bits

7	6	5	4	3	2	1	0
SYSCFG 50h			**PMU Control Register 4**				**Default = 00h**
Software start SMI: 0 = Clear SMI 1 = Start SMI							
SYSCFG 59h			**PMU Event Register 2**				**Default = 00h**
Allow software SMI: 0 = Disable 1 = Enable							

4.17.2 SMI Event Generation

The registers shown in Table 4-155 control the events that are allowed to generate an SMI. The programming occurs as follows: time-out, access, and interrupt events must be programmed to generate a PMI, then the PMI event must be enabled to generate the SMI signal; finally, SMIs are globally unmasked to allow full operation.

4.17.2.1 Time-out Event Generation of SMI

For time-out events, simply loading a non-zero timer value and generating a dummy access presets PMI generation on the next time-out. Refer to the section titled "Timers" on page 172 for information on programming the timers.

4.17.2.2 Access Event Generation of SMI

Access events can be programmed to generate an SMI. FireStar classifies accesses as "Current Access" or "Next Access" depending on whether the timer associated with that access range is still running or has timed out.

- Next Access - Occurs after a time-out, the first time software attempts to access the range that caused the time-out. The Next Access feature provides a way for I/O accesses to a peripheral whose timer has timed out to cause an SMI so that the peripheral can be powered up before the access takes place. Next Access can also restart system clocks when the system is in the Doze mode.
- Current Access - Occurs any time this feature is enabled for a range, whether or not the device has timed out. The Current Access PMI can be programmed to cause an SMI, but cannot provide any automatic means of controlling system clocks.

If both the Current Access and Next Access features are enabled for an event and the timer has timed out, an access will only cause a single SMI. Since both access types use the same PMI#, clearing either one clears both events.

The I/O blocking bit, SYSCFG DBh[7], operates as follows. This selection allows the I/O access that causes a Next Access PMI to be either blocked (if the peripheral is turned off, for example) or passed through. DBh[7] = 1 means the I/O will not be blocked; DBh[7] = 0 means the I/O on Next Access will be blocked and the CPU must be programmed to restart the I/O command if desired. The feature defaults to "blocked".

Table 4-155 Current and Next Access Registers

7	6	5	4	3	2	1	0
SYSCFG 5Bh if AEh[7] = 0			**PMU Event Register 4**				**Default = 00h**
Reserved	Global SMI control: 0 = Allow 1 = Mask	Reserved		GNR1 Next Access PMI#15: 0 = Disable 1 = Enable	KBD Next Access PMI#14: 0 = Disable 1 = Enable	DSK Next Access PMI#13: 0 = Disable 1 = Enable	LCD Next Access PMI#12: 0 = Disable 1 = Enable
SYSCFG 5Bh if AEh[7] = 1			**PMU Event Register 4A**				**Default = 00h**
Reserved				GNR5 Next Access PMI#15: 0 = Disable 1 = Enable	Reserved		
SYSCFG DBh if AEh[7] = 0			**Next Access Event Generation Register 2**				**Default = 00h**
I/O blocking control: 0 = Block I/O on Next Access trap 1 = Unblock	SMI on cool-down clocking entry/exit: 0 = Disable 1 = Enable	EPMI3# pin polarity: 0 = Active high 1 = Active low	EPMI2# pin polarity: 0 = Active high 1 = Active low	HDU_ ACCESS PMI#23 on Next Access: 0 = No 1 = Yes	COM2_ ACCESS PMI#22 on Next Access: 0 = No 1 = Yes	COM1_ ACCESS PMI#21 on Next Access: 0 = No 1 = Yes	GNR2_ ACCESS PMI#20 on Next Access: 0 = No 1 = Yes
SYSCFG DBh if AEh[7] = 1			**Next Access Event Generation Register 2A**				**Default = 00h**
Reserved							GNR6_ ACCESS PMI#20 on Next Access: 0 = No 1 = Yes

Table 4-155 Current and Next Access Registers (cont.)

7	6	5	4	3	2	1	0
SYSCFG DEh if AEh[7] = 0			**Current Access Event Generation Register 1**				**Default = 00h**
HDU_ ACCESS PMI#23 on Current Access: 0 = No 1 = Yes	COM2_ ACCESS PMI#22 on Current Access: 0 = No 1 = Yes	COM1_ ACCESS PMI#21 on Current Access: 0 = No 1 = Yes	GNR2_ ACCESS PMI#20 on Current Access: 0 = No 1 = Yes	GNR1_ ACCESS PMI#15 on Current Access: 0 = No 1 = Yes	KBD_ ACCESS PMI#14 on Current Access: 0 = No 1 = Yes	DSK_ ACCESS PMI#13 on Current Access: 0 = No 1 = Yes	LCD_ ACCESS PMI#12 on Current Access: 0 = No 1 = Yes
SYSCFG DEh if AEh[7] = 1			**Current Access Event Generation Register 1A**				**Default = 00h**
Reserved			GNR6_ ACCESS PMI#20 on Current Access: 0 = No 1 = Yes	Reserved			
SYSCFG E9h if AEh[7] = 0			**PMU Event Register 7**				**Default = 00h**
GNR4_TIMER PMI#30 GNR4_ACCESS PMI#32: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		GNR3_TIMER PMI#29 GNR3_ACCESS PMI#31: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		GNR4_ ACCESS PMI#32 on Current Access: 0 = No 1 = Yes	GNR4_ ACCESS PMI#32 on Next Access: 0 = No 1 = Yes	GNR3_ ACCESS PMI#31 on Current Access: 0 = No 1 = Yes	GNR3_ ACCESS PMI#31 on Next Access: 0 = No 1 = Yes
SYSCFG E9h if AEh[7] = 1			**PMU Event Register 7A**				**Default = 00h**
GNR8_TIMER PMI#30 GNR8_ACCESS PMI#32: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		GNR7_TIMER PMI#29 GNR7_ACCESS PMI#31: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		GNR8_ ACCESS PMI#32 on Current Access: 0 = No 1 = Yes	GNR8_ ACCESS PMI#32 on Next Access: 0 = No 1 = Yes	GNR7_ ACCESS PMI#31 on Current Access: 0 = No 1 = Yes	GNR7_ ACCESS PMI#31 on Next Access: 0 = No 1 = Yes

4.17.2.3 Interrupt Event Generation of SMI

Asynchronous events from peripheral devices requesting service from the CPU are known as interrupt events. Interrupts in this context include both the traditional AT architecture IRQs and additional inputs known as external power management inputs. For FireStar logic, the desired interrupts are all grouped into a single event called INTRGRP. INTRGRP can then be enabled to cause an SMI.

If it is desired to generate an SMI from the INTRGRP event, setting SYSCFG 57h[6] = 1 will allow any of the selected interrupt events to generate PMI#6. Once in the SMI handler, the SMM code can read the SYSCFG 64h and A4h to determine which of the interrupt(s) caused the event. The IRQs will remain latched for reading in these registers until PMI#6 is cleared, at which time any latched sources are cleared. The INTRGRP IRQ Select Registers are shown in Table 4-156.

4.17.2.4 DRQ Event Generation of SMI

FireStar allows activity on the DRQ pins to generate an SMI. The SMI takes place before the DMA transfer occurs, allowing SMM code to emulate or modify the operation. Writing the bit to clear the PMI allows any pending DMA operation to take place immediately. Note that there are certain latency limitations for DMA operations. For example, floppy disk DMA transfers generally must be serviced within 14μs from receipt of DRQ2 in order to avoid an overrun condition. Entry into SMM requires a considerable amount of time in itself. Therefore, SMM routines that trap DMA accesses must be structured concisely so that the DMA cycle is allowed to occur before the latency limit exceeded. Table 4-157 shows which register bits apply to this application.

Table 4-156 INTRGRP IRQ Select Register Bits

7	6	5	4	3	2	1	0
SYSCFG 57h			**PMU Control Register 5**				**Default = 08h**
	INTRGRP generates PMI#6: 0 = Disable 1 = Enable						
SYSCFG 64h			**INTRGRP IRQ Select Register 1**				**Default = 00h**
IRQ14: 0 = Disable 1 = Enable	IRQ8: 0 = Disable 1 = Enable	IRQ7: 0 = Disable 1 = Enable	IRQ6: 0 = Disable 1 = Enable	IRQ5: 0 = Disable 1 = Enable	IRQ4: 0 = Disable 1 = Enable	IRQ3: 0 = Disable 1 = Enable	IRQ1: 0 = Disable 1 = Enable
SYSCFG A4h			**INTRGRP IRQ Select Register 2**				**Default = 00h**
	IRQ15: 0 = Disable 1 = Enable	IRQ13: 0 = Disable 1 = Enable	IRQ12: 0 = Disable 1 = Enable	IRQ11: 0 = Disable 1 = Enable	IRQ10: 0 = Disable 1 = Enable	IRQ9: 0 = Disable 1 = Enable	IRQ0: 0 = Disable 1 = Enable

Table 4-157 DMA DRQ Trap SMI Register Bits

7	6	5	4	3	2	1	0
SYSCFG D6h			**PMU Control Register 10**				**Default = 00h**
	DMA trap PMI#28 SMI: 0 = Disable 1 = Enable						
SYSCFG DDh			**PMU SMI Source Register 4 (Write 1 to Clear)**				**Default = 00h**
		PMI#37, DMA_ ACCESS: 0 = Inactive 1 = Active	PMI#28, DMA Request: 0 = Inactive 1 = Active				

4.17.3 Enabling of Events to Generate SMI

The registers listed in Table 4-158 allow PMI events that are enabled to generate timer time-outs, accesses, and interrupts to cause SMIs. Before setting the SMI Event Enable Registers, time-outs, accesses, and interrupts must be individually enabled to generate PMI events as follows.

- For time-out events, loading a non-zero timer value and generating a dummy access presets PMI generation on the next time-out.
- For Current Access events, the appropriate current access enable bit must be set to preset PMI generation on the following access.
- For Next Access events, the appropriate next access enable bit must be set. Then, a valid time-out must take place to preset PMI generation on the following access.
- For interrupt events, the corresponding INTRGRP bit must be set and INTRGRP must be enabled to generate PMI#6. Then, on any enabled interrupt PMI#6 will occur.

Only after all desired PMI events have been enabled should the PMI be enabled to generate an SMI through the register set below. Setting SYSCFG 5Bh[6] = 1 then unmasks all the SMIs previously enabled.

Note that a Resume event can be enabled to generate PMI#6. Refer to the “Suspend and Resume” section for details on enabling resume events.

4.17.3.1 PMI#25 Triggers

The PMI#25 event is shared by both EPMI3# and the thermal management unit. SYSCFG D9h[3:2] enable SMI for EPMI3# only. SYSCFG DBh[6] enables SMI only for cool-down clocking entry and exit.

Table 4-158 SMI Event Enable Registers

7	6	5	4	3	2	1	0
SYSCFG 58h			**PMU Event Register 1**				**Default = 00h**
LOWBAT PMI#3 SMI: 00 = Disable 11 = Enable		EPMI1# PMI#2 SMI: 00 = Disable 11 = Enable		EPMI0# PMI#1 SMI: 00 = Disable 11 = Enable		LLOWBAT PMI#0 SMI: 00 = Disable 11 = Enable	
SYSCFG 59h			**PMU Event Register 2**				**Default = 00h**
		Resume INTRGRP PMI#6, Suspend PMI#7 SMI: 00 = Disable 11 = Enable		R_TIMER PMI#5 SMI: 00 = Disable 11 = Enable		IDLE_TIMER PMI#4 SMI: 00 = Disable 11 = Enable	
SYSCFG 5Ah if AEh[7] = 0			**PMU Event Register 3**				**Default = 00h**
GNR1_TIMER PMI#11 GNR1_ACCESS PMI#15: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		KBD_TIMER PMI#10 KBD_ACCESS PMI#14: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		DSK_TIMER PMI#9 DSK_ACCESS PMI#13: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		LCD_TIMER PMI#8 LCD_ACCESS PMI#12: 00 = Disable 01 = Reserved 10 = Reserved 11 = SMI	
SYSCFG 5Ah if AEh[7] = 1			**PMU Event Register 3A**				**Default = 00h**
GNR5_TIMER PMI: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		Reserved					
SYSCFG 5Bh if AEh[7] = 0			**PMU Event Register 4**				**Default = 00h**
	Global SMI control: 0 = Allow 1 = Mask						

Table 4-158 SMI Event Enable Registers (cont.)

7	6	5	4	3	2	1	0
SYSCFG D8h if AEh[7] = 0			**PMU Event Register 5**				**Default = 00h**
HDU_TIMER PMI#19 HDU_ACCESS PMI#23: 00 = Disable 01 = Reserved 01 = Reserved 11 = SMI		COM2_TIMER PMI#18 COM2_ACCESS PMI#22: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		COM1_TIMER PMI#17 COM1_ACCESS PMI#21: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		GNR2_TIMER PMI#16 GNR2_ACCESS PMI#20: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI	
SYSCFG D8h if AEh[7] = 1			**PMU Event Register 5A**				**Default = 00h**
Reserved						GNR6_TIMER PMI#16 GNR6_ACCESS PMI#20: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI	
SYSCFG D9h			**PMU Event Register 6**				**Default = 00h**
DOZE_TIMER PMI#27 SMI: 00 = Disable 01 = Enable DOZE_0 10 = Enable DOZE_1 11 = Enable both		RINGI PMI#26 SMI: 00 = Disable 11 = Enable		EPMI3# cool-down clocking PMI#25 SMI: 00 = Disable 11 = Enable		EPMI2# PMI#24 SMI: 00 = Disable 11 = Enable	
SYSCFG DBh if AEh[7] = 0			**Next Access Event Generation Register 2**				**Default = 00h**
	SMI on cool-down clocking entry/exit: 0 = Disable 1 = Enable						
SYSCFG E9h if AEh[7] = 0			**PMU Event Register 7**				**Default = 00h**
GNR4_TIMER PMI#30 GNR4_ACCESS PMI#32: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		GNR3_TIMER PMI#29 GNR3_ACCESS PMI#31: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI					
SYSCFG E9h if AEh[7] = 1			**PMU Event Register 7A**				**Default = 00h**
GNR8_TIMER PMI#30 GNR8_ACCESS PMI#32: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		GNR7_TIMER PMI#29 GNR7_ACCESS PMI#31: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI					

4.17.4 Servicing an SMI

The register set shown in Table 4-159 is used by SMM code to enable system events to cause SMIs, to determine the events that caused an active SMI, and to clear the events. Determining the source of the SMI is an uncomplicated procedure.

1. Upon entry to SMM, read the SYSCFG 5Ch, 5Dh, DCh, DDh, and EAh. Any non-zero bits indicate PMI sources. More than one can be active.
2. The PMI number will indicate the source of the service request. If the PMI#6 is generated, also read SYSCFG 64h and A4h (described earlier in Section 4.17.2.3, "Interrupt Event Generation of SMI") to determine which IRQ line was responsible for the event.
3. Service the events in the order desired; upon completion of each service, write a 1 back to the event source register bit to clear that event. Continue in this manner until all events are serviced and all the service registers are clear.
4. Issue the proper CPU instruction to return from SMM operation. If any events are still pending, most CPUs will immediately re-enter SMM.

Table 4-159 SMI Service Registers

7	6	5	4	3	2	1	0
SYSCFG 5Ch			**PMI SMI Source Register 1 (Write 1 to Clear)**				**Default = 00h**
PMI#7, Suspend: 0 = Not Active 1 = Active	PMI#6, Resume or INTRGRP: 0 = Not Active 1 = Active	PMI#5, R_TIMER time-out: 0 = Not Active 1 = Active	PMI#4, IDLE_TIMER time-out: 0 = Not Active 1 = Active	PMI#3, LOWBAT: 0 = Not Active 1 = Active	PMI#2, EPMI1#: 0 = Not Active 1 = Active	PMI#1, EPMI0#: 0 = Not Active 1 = Active	PMI#0, LLOWBAT: 0 = Not Active 1 = Active
SYSCFG 5Dh if AEh[7] = 0			**PMI SMI Source Register 2 (Write 1 to Clear)**				**Default = 00h**
PMI#15, GNR1_ ACCESS: 0 = None 1 = Active	PMI#14, KBD_ACCESS: 0 = Not Active 1 = Active	PMI#13, DSK_ACCESS: 0 = Not Active 1 = Active	PMI#12, LCD_ACCESS: 0 = Not Active 1 = Active	PMI#11, GNR1_TIMER: 0 = Not Active 1 = Active	PMI#10, KBD_TIMER: 0 = Not Active 1= Active	PMI#9, DSK_TIMER: 0 = Not Active 1 = Active	PMI#8, LCD_TIMER: 0 = Not Active 1 = Active
SYSCFG 5Dh if AEh[7] = 1			**PMI SMI Source Register 2A (Write 1 to Clear)**				**Default = 00h**
PMI#15, GNR5_ ACCESS: 0 = None 1 = Active	Reserved			PMI#11 GNR5_TIMER: 0 = None 1 = Active	Reserved		
SYSCFG DCh if AEh[7] = 0			**PMU SMI Source Register 3 (Write 1 to Clear)**				**Default = 00h**
PMI#23, HDU_ ACCESS: 0 = Inactive 1 = Active	PMI#22, COM2_ ACCESS: 0 = Inactive 1 = Active	PMI#21, COM1_ ACCESS: 0 = Inactive 1 = Active	PMI#20, GNR2_ ACCESS: 0 = Inactive 1 = Active	PMI#19, HDU_ TIMER: 0 = Inactive 1 = Active	PMI#18, COM2_ TIMER: 0 = Inactive 1 = Active	PMI#17, COM1_ TIMER: 0 = Inactive 1 = Active	PMI#16, GNR2_ TIMER: 0 = Inactive 1 = Active
SYSCFG DCh if AEh[7] = 1			**PMU SMI Source Register 3A (Write 1 to Clear)**				**Default = 00h**
Reserved			PMI#20, GNR6_ ACCESS: 0 = Clear 1 = Active	Reserved			PMI#16, GNR6_ TIMER: 0 = Clear 1 = Active

Table 4-159 SMI Service Registers (cont.)

7	6	5	4	3	2	1	0
SYSCFG DDh			**PMU SMI Source Register 4 (Write 1 to Clear)**				**Default = 00h**
PMI#39, PCI retry limit: 0 = Inactive 1 = Active	PMI#38, CISA/PCI IRQ driveback trap: 0 = Inactive 1 = Active	PMI#37, DMA_ ACCESS: 0 = Inactive 1 = Active	PMI#28, DMA Request: 0 = Inactive 1 = Active	PMI#27, DOZE_ TIMER: 0 = Inactive 1 = Active	PMI#26, RINGI: 0 = Inactive 1 = Active	PMI#25, EPMI3# pin/ cool-down clocking: 0 = Inactive 1 = Active	PMI#24, EPMI2# pin: 0 = Inactive 1 = Active
SYSCFG DDh - FS ACPI Version			**PMU SMI Source Register 4 (Write 1 to Clear)**				**Default = 00h**
PMI#39, ACPI SMI: 0 = Inactive 1 = Active	PMI#38, CISA/PCI IRQ driveback trap: 0 = Inactive 1 = Active	PMI#37, DMA_ ACCESS: 0 = Inactive 1 = Active	PMI#28, DMA Request: 0 = Inactive 1 = Active	PMI#27, DOZE_ TIMER: 0 = Inactive 1 = Active	PMI#26, RINGI: 0 = Inactive 1 = Active	PMI#25, EPMI3# pin/ cool-down clocking: 0 = Inactive 1 = Active	PMI#24, EPMI2# pin: 0 = Inactive 1 = Active
SYSCFG EAh if AEh[7] = 0			**PMU SMI Source Register 5 (Write 1 to Clear)**				**Default = 00h**
PMI#36, Serial IRQ trap: 0 = Inactive 1 = Active	PMI#35, APM Doze exit: 0 = Inactive 1 = Active	PMI#34, Hot docking time-out SMI: 0 = Inactive 1 = Active	PMI#33, H/W DOZE_ TIMER reload (on Doze exit): 0 = Inactive 1 = Active	PMI#32, GNR4_ ACCESS 0 = Inactive 1 = Active	PMI#31, GNR3_ ACCESS 0 = Inactive 1 = Active	PMI#30, GNR4_ TIMER 0 = Inactive 1 = Active	PMI#29, GNR3_ TIMER 0 = Inactive 1 = Active
SYSCFG EAh if AEh[7] = 1			**PMU SMI Source Register 8 (Write 1 to Clear)**				**Default = 00h**
Reserved				PMI#32, GNR8_ ACCESS 0 = Inactive 1 = Active	PMI#31, GNR7_ ACCESS 0 = Inactive 1 = Active	PMI#30, GNR8_ TIMER 0 = Inactive 1 = Active	PMI#29, GNR7_ TIMER 0 = Inactive 1 = Active

4.17.4.1 PMI Source Register Details

SYSCFG 5Ch, 5Dh, DCh, DDh, and EAh indicate the SMI source. When a PMI event occurs, the corresponding bit will be set to 1 and the SMI# signal will then be generated. In the SMI service routine, SMM code must check these registers for the PMI source(s) and then clear them. Otherwise, for all but the EPMI pins, the latched PMI source will generate SMI# continuously. SMI code normally clears only one event at a time to keep track of the events as they are serviced, but all events can be cleared at once if desired. Note that clearing SYSCFG 5Ch[6] will clear SYSCFG 5Ch[7] also.

Refer to Section 4.15.5, "Suspend and Resume" for information on PMI#6 when it is used to indicate a resume event.

4.17.4.2 EPMI Pin PMI Sources

The EPMI[1:0]# pins' PMI source indicator bits behave a little differently than the rest of the PMI source indicator bits. For PMI#1 and PMI#2, the EPMI[1:0]# inputs are **not** latched by default, so SYSCFG 5Ch[2:1] are not latched. Therefore, an external device could trigger an SMI by toggling one of the EPMI[1:0]# lines, but if the device returns the EPMI line to its inactive state before SMM code reads SYSCFG 5Ch[2:1], the code would not be able to recognize the event that triggered the SMI. Likewise, an EPMI[1:0]# edge could initiate a resume from suspend mode, but then would not be recognized if the EPMI pin went to its inactive state.

SYSCFG A1h[0] is provided to allow EPMI[1:0]# to be latched like other PMIs. If SYSCFG A1h[0] is written to 1, EPMI[1:0]# events will be latched at SYSCFG 5Ch[2:1]. Writing a 1 into the active bit(s) then clears the PMI.

For PMI#24 and PMI#25, the EPMI[3:2]# inputs are always latched, regardless of the A1h[0] setting.

4.17.5 I/O SMI Trap Indication

FireStar provides a means for SMM code to determine the I/O port whose access caused the SMI, a bit to indicate whether the access was a read or a write, as well as the write data with SBHE# status for write instructions. The registers that provide the above data are shown in Table 4-160.

Table 4-160 I/O SMI Trap Indication Registers

7	6	5	4	3	2	1	0
SYSCFG D6h			**PMU Control Register 10**				**Default = 00h**
			APM doze exit PMI#35: 0 = Disable 1 = Enable	SBHE# status trap (RO)	I/O port access trapped (RO): 0 = I/O read 1 = I/O write	Access trap bit A9 (RO)	Access trap bit A8 (RO)
SYSCFG D7h			**Access Port Address Register 1**				**Default = 00h**
Access trap address bits A[7:0]: - These bits, along with SYSCFG D6h[1:0] and SYSCFG EBh[7:0] provide the 16-bit address of the port access that caused the SMI trap. - SYSCFG D6h[2] indicates whether an I/O read or an I/O write access was trapped. - SYSCFG D6h[3] gives the status of the SBHE# signal for the I/O instruction that was trapped.							
SYSCFG EBh			**Access Port Address Register 2**				**Default = 00h**
Reserved		Access trap address bits A[15:10]: These bits along with SYSCFG D6h[1:0] and D7h[7:0] provide the 16-bit address of the port access that caused the SMI trap. D6h[2] indicates whether an I/O read or an I/O write access was trapped. D6h[3] gives the status of the SBHE# signal for the I/O instruction that was trapped.					
SYSCFG ECh			**Write Trap Register 1 (RO)**				**Default = 00h**
I/O write data trap[15:8]: - Along with SYSCFG EDh[7:0], this register provides the 16-bit write data for trapped I/O write instructions.							
SYSCFG EDh			**Write Trap Register 2 (RO)**				**Default = 00h**
I/O write data trap[7:0]: - Along with SYSCFG ECh[7:0], this register provides the 16-bit write data for trapped I/O write instructions							

4.17.6 Trapping with NMI Instead of SMM

Many system events require more immediate service than is possible with the current SMI scheme, which takes approximately 4µs to enter SMM and another 4µs to exit. Therefore, FireStar offers the possibility of selectively diverting the PMI event from generating SMI with NMI. The drawback to this scheme is the lack of protection involved. The interrupt vector in low memory cannot be write-protected. Hence, using NMI trapping is not as secure as SMI trapping.

Note that the NMI Trap Enable bit must not be set if an SMI is enabled for this PMI event. Table 4-161 shows the NMI Trap Enable registers.

Table 4-161 NMI Trap Enable Bits

7	6	5	4	3	2	1	0
SYSCFG 38h			**NMI Trap Enable Register 1**				**Default = 00h**
PMI#7 NMI: 0 = Disable 1 = Enable	PMI#6 NMI: 0 = Disable 1 = Enable	PMI#5 NMI: 0 = Disable 1 = Enable	PMI#4 NMI: 0 = Disable 1 = Enable	PMI#3 NMI: 0 = Disable 1 = Enable	PMI#2 NMI: 0 = Disable 1 = Enable	PMI#1 NMI: 0 = Disable 1 = Enable	PMI#0 NMI: 0 = Disable 1 = Enable
SYSCFG 39h			**NMI Trap Enable Register 2**				**Default = 00h**
PMI#15 NMI: 0 = Disable 1 = Enable	PMI#14 NMI: 0 = Disable 1 = Enable	PMI#13 NMI: 0 = Disable 1 = Enable	PMI#12 NMI: 0 = Disable 1 = Enable	PMI#11 NMI: 0 = Disable 1 = Enable	PMI#10 NMI: 0 = Disable 1 = Enable	PMI#9 NMI: 0 = Disable 1 = Enable	PMI#8 NMI: 0 = Disable 1 = Enable
SYSCFG 3Ah			**NMI Trap Enable Register 3**				**Default = 00h**
PMI#23 NMI: 0 = Disable 1 = Enable	PMI#22 NMI: 0 = Disable 1 = Enable	PMI#21 NMI: 0 = Disable 1 = Enable	PMI#20 NMI: 0 = Disable 1 = Enable	PMI#19 NMI: 0 = Disable 1 = Enable	PMI#18 NMI: 0 = Disable 1 = Enable	PMI#17 NMI: 0 = Disable 1 = Enable	PMI#16 NMI: 0 = Disable 1 = Enable
SYSCFG 3Bh			**NMI Trap Enable Register 4**				**Default = 00h**
PMI#31 NMI: 0 = Disable 1 = Enable	PMI#30 NMI: 0 = Disable 1 = Enable	PMI#29 NMI: 0 = Disable 1 = Enable	PMI#28 NMI: 0 = Disable 1 = Enable	PMI#27 NMI: 0 = Disable 1 = Enable	PMI#26 NMI: 0 = Disable 1 = Enable	PMI#25 NMI: 0 = Disable 1 = Enable	PMI#24 NMI: 0 = Disable 1 = Enable
SYSCFG 3Ch			**NMI Trap Enable Register 5**				**Default = 00h**
Reserved		PMI#37 NMI: 0 = Disable 1 = Enable	PMI#36 NMI: 0 = Disable 1 = Enable	PMI#35 NMI: 0 = Disable 1 = Enable	PMI#34 NMI: 0 = Disable 1 = Enable	PMI#33 NMI: 0 = Disable 1 = Enable	PMI#32 NMI: 0 = Disable 1 = Enable

4.18 Utility Registers

The registers below provide SMM code with a general purpose storage region and a means of generating warning beeps on the system speaker without modifying the ISA-compatible I/O ports.

Table 4-162 Utility Registers

7	6	5	4	3	2	1	0
SYSCFG 51h			**Beeper Control Register**				**Default = 00h**
Reserved						Beeper control: 00 = No Action 01 = 1kHz 10 = Off 11 = 2kHz	
SYSCFG 52h			**Scratchpad Register 1**				**Default = 00h**
General purpose storage byte: - For CISA Configuration Cycles: Data phase information, low byte							
SYSCFG 53h			**Scratchpad Register 2**				**Default = 00h**
General purpose storage byte. - For CISA Configuration Cycles: Data phase information, high byte							
SYSCFG 6Ch			**Scratchpad Register 3**				**Default = 00h**
General purpose storage byte - For CISA Configuration Cycles: Address phase 1 information, low byte							
SYSCFG 6Dh			**Scratchpad Register 4**				**Default = 00h**
General purpose storage byte - For CISA Configuration Cycles: Address phase 1 information, high byte							
SYSCFG 6Eh			**Scratchpad Register 5**				**Default = 00h**
General purpose storage byte - For CISA Configuration Cycles: Address phase 2 information, low byte							
SYSCFG 6Fh			**Scratchpad Register 6**				**Default = 00h**
General purpose storage byte - For CISA Configuration Cycles: Address phase 2 information, high byte							

4.19 Hot Docking Feature

The "hot" attachment of a docking station to a notebook computer requires the computer to have certain capabilities that are listed below:

1. A mechanism to sense the beginning and the end of docking.
2. The ability to tristate the ISA and PCI buses when docking is in progress and to not generate bus cycles during that period.
3. The capability of either continuing with normal operation, or of generating an SMI if the end of docking is not sensed within a certain time period.

The docking station also needs to have the capability of indicating the start of docking and the capability to indicate the completion of docking. This is usually accomplished by using special dock connectors that have "early" and "late" connections. The male connector is on the docking station and has several long pins. During insertion, these pins make contact with their counterparts on the notebook earlier than the other pins. One of these pins (HDI - Hot Docking Indicator) may be asserted by the docking station and can be used to indicate the beginning of insertion. Later on, when all pins make contact, the HDI pin may be deasserted to indicate the completion of docking, following which the notebook may start driving the ISA bus again. Referring to Figure 4-42, traversal time may be defined as the time taken for the shorter pins to make contact after the longer pins have made contact. The ISA bus signals have to be tristated after the HDI pin is detected as active and before the shorter pins make contact. This is not expected to be a problem because traversal time is usually of the order of a few milliseconds, whereas the longest back-to-back ISA cycles are of the order of a few microseconds; hot docking may therefore be detected and the ISA bus tristated well within the traversal time period.

FireStar implements this feature by making use of one of the EPMI# inputs as the HDI pin and a programmable time-out counter that is loaded with a value that is an estimate of the traversal time. Any one of the EPMI# inputs may be programmed to perform the HDI function. An active EPMI# input is an indication to the chipset that docking is in progress.

On the occurrence of an active signal on the selected EPMI# input, the time-out counter is started, as shown in Figure 4-43. FireStar's logic then causes the CPU to stop operation after the current cycle is completed by placing it on hold to ensure that another ISA bus cycle is not started while docking is in progress. After the system completes the current ISA bus cycle, which could be in the order of a few microseconds for certain ISA bus cycles, the ISA bus signals are tristated, and remain tristated till either the time-out counter runs out, or the HDI input is deasserted, whichever occurs earlier. The HDI input is expected to be deasserted within the time-out period. If the HDI input is deasserted within the time-out period as shown in Figure 4-43, it indicates that docking was completed within the expected time of insertion, and that the ISA bus may be driven again. If it is not deasserted within the time-out period as shown in Figure 4-44, it may be construed that docking was not completed in the expected time. In this situation, the option of either generating an SMI or ignoring the time-out and driving the ISA bus again is provided. Figure 4-44 shows the case where an SMI is generated due to the time-out period elapsing.

Note that the PCI bus is automatically disabled, and its signals tristated, when the ISA bus is tristated. Therefore, it may be possible to hot dock on the PCI bus as well as the ISA bus.

Figure 4-42 Insertion Times

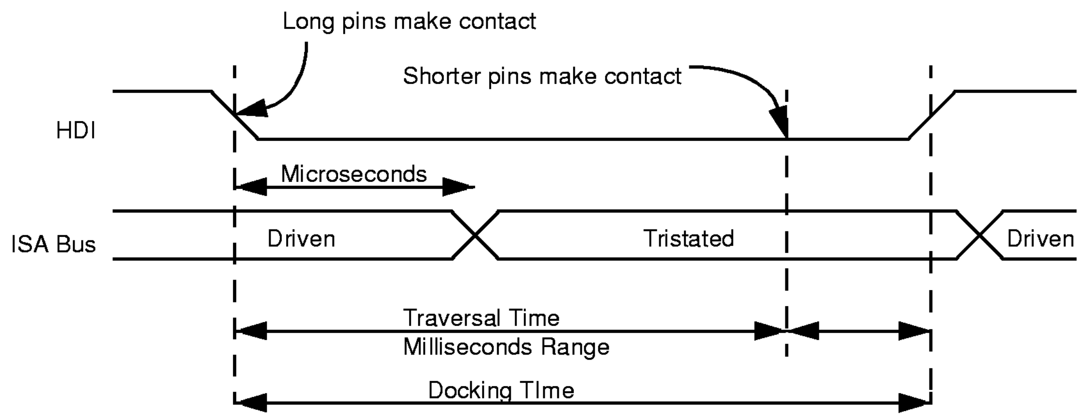


Figure 4-43 HDI Input Deasserted Within Time-out Period

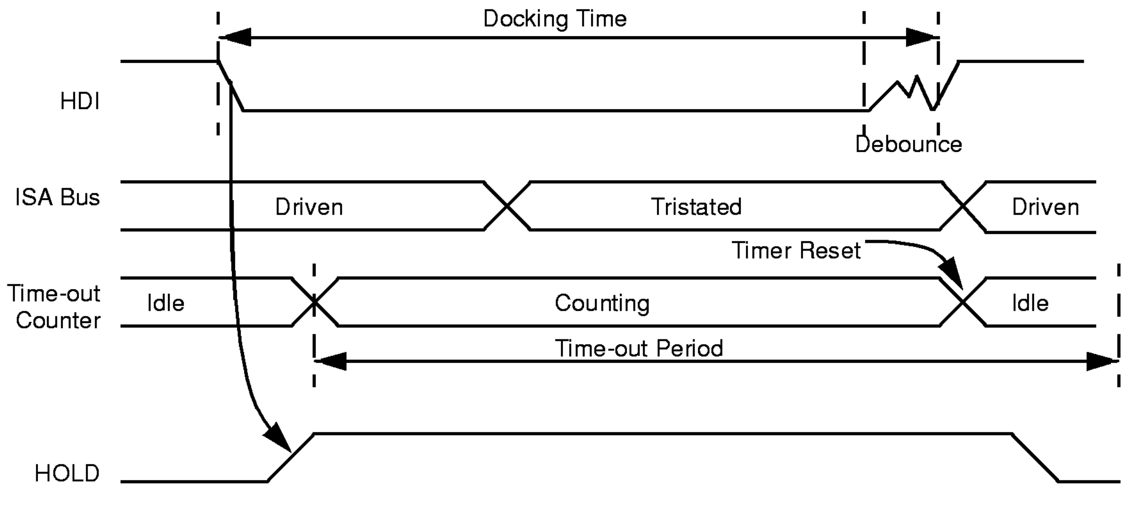
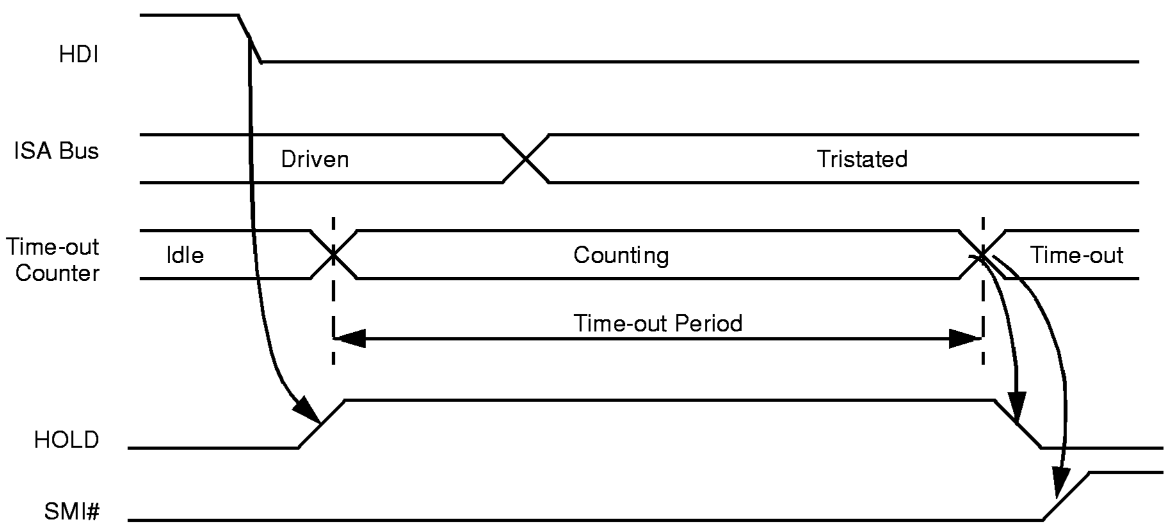


Figure 4-44 HDI Input Not Deasserted Within Time-out Period (SMI generated on time-out)



4.19.1 Initialization Procedure:

1. Select the EPMI# input on which HDI will be provided by appropriately setting SYSCFG F0h[1:0].
2. Set SYSCFG EFh[6:5] if debouncing is required on the selected EPMI# pin. It is recommended that debouncing be enabled.
3. Select the polarity of the HDI input by appropriately setting SYSCFG 40h[2], or 40h[1], or DBh[5], or DBh[4], depending on which EPMI# input is selected as the HDI input.
4. Docking may be done when the system is in suspend. If so desired, disable the capacity of the selected EPMI# to generate a resume by setting SYSCFG B1h[7], or B1h[6], or 6Ah[7], or 6Ah[6] to 0. If a resume operation on docking is desired, the appropriate bit must be set to 1.
5. Set the time-out period by programming SYSCFG EFh[2:0]. If the system is in suspend, a default time-out of 1ms generated from the 32kHz clock is used to override this setting.
6. If an SMI is to be generated on time-out, set SYSCFG EFh[4].
7. Enable the capacity of the selected EPMI# input to generate an SMI by setting SYSCFG 58h[5:4], or 58h[3:2], or D9h[3:2], or D9h[1:0] to 11b.
8. Enable hot docking by setting SYSCFG EFh[7] to 1.
9. Finally, enable the global SMI generation control by setting SYSCFG 5Bh[6] to 0.

Table 4-163 shows the register bits associated with hot docking.

Table 4-163 Hot Docking Control Register Bits

7	6	5	4	3	2	1	0
SYSCFG EFh			**Hot Docking Control Register 1**				**Default = 00h**
Hot docking enable: 0 = Disable 1 = Enable (Default)	HDI input debounce rate: 00 = 100μs 01 = 512μs 10 = 1ms 11 = 2ms		HDI active level: 0 = Active high 1 = Active low Also see SYSCFG AAh[0]	HDI SMI: 0 = No SMI on time-out (Default) 1 = Generate SMI on time-out	HDI time-out period: 000 = 1ms 001 = 8ms 010 = 64ms 011 = 256ms 100 = 512ms 101 = 2s 110 = 8s 111 = 16s		
SYSCFG AAh			**Thermal Management Register 6**				**Default = 00h**
							HDI input: 0 = HDI 1 = EPMI indicated by SYSCFG F0h[1:0]
SYSCFG F0h			**Hot Docking Control Register 2**				**Default = 00h**
						EPMI trigger for HDI: 00 = EPMI0# 01 = EPMI1# 10 = EPMI2# 11 = EPMI3# Also see SYSCFG AAh[0]	
SYSCFG 40h			**PMU Control Register 1**				**Default = 00h**
					EPMI1# polarity: 0 = Active high 1 = Active low	EPMI0# polarity: 0 = Active high 1 = Active low	

Table 4-163 Hot Docking Control Register Bits (cont.)

7	6	5	4	3	2	1	0
SYSCFG DBh if AEh[7] = 0			**Next Access Event Generation Register 2**				**Default = 00h**
		External EPMI3# pin polarity: 0 = Active high 1 = Active low	External EPMI2# pin polarity: 0 = Active high 1 = Active low				
SYSCFG B1h			**RSMGRP IRQ Register 2**				**Default = 00h**
EPMI3# Resume: 0 = Disable 1 = Enable	EPMI2# Resume: 0 = Disable 1 = Enable						
SYSCFG 6Ah			**RSGGRP IRQ Register 1**				**Default = 00h**
EPMI1# Resume: 0 = Disable 1 = Enable	EPMI0# Resume: 0 = Disable 1 = Enable						
SYSCFG 58h			**PMU Event Register 1**				**Default = 00h**
		EPMI1# PMI#2 SMI: 00 = Disable 11 = Enable		EPMI0# PMI#1 SMI: 00 = Disable 11 = Enable			
SYSCFG D9h			**PMU Event Register 6**				**Default = 00h**
				EPMI3# cool-down clocking PMI#25 SMI: 00 = Disable 11 = Enable		EPMI2# PMI#24 SMI: 00 = Disable 11 = Enable	
SYSCFG 5Bh if AEh[7] = 0			**PMU Event Register 4**				**Default = 00h**
	Global SMI control: 0 = Allow 1 = Mask						

5.0 Register Descriptions

There are three broad classes of FireStar configuration registers spaces:

1) PCI Configuration Register Space
2) System Control Register Space
3) I/O Register Space

Table 5-1 details the locations and access mechanisms for registers located within these register spaces.

Notes:
1. Bits/registers that are new or have changed (between Data Book Rev 0.2 and Data Book Rev 1.0), are underlined and denoted with a change bar in the margin.
2. All bits/registers are read/write and their default value is 0 unless otherwise specified.
3. All reserved bits/registers **MUST** be written to 0 unless otherwise specified.

Table 5-1 Register Locations and Access Mechanisms

Register Space	Location	Access Mechanism	Reference Section
System Control	SYSCFG 00h-FFh	Index loaded in 022h, Data to/from index through 024h	Section 5.1, SYSCFG Register Space
PCI Configuration	PCIDV0 00h-FFh	Through PCI Configuration Mechanism #1 as: Bus #0, Device #0, Function #0	Section 5.2, PCIDV0 Register Space
	PCIDV1 00h-FFh	Through PCI Configuration Mechanism #1 as: Bus #0, Device #1, Function #0	Section 5.3, PCIDV1 Register Space
	PCIIDE 00h-47h	Through PCI Configuration Mechanism #1 as: Bus #0, Device #14h, Function #0, or Bus #0, Device #1, Function #1	Section 5.4, IDE Register Space
I/O Register	Port 000h-FFFFh	CPU Direct I/O R/W	Section 5.5, I/O Register Space

The following briefly describes how to access FireStar and PCI devices. FireStar uses PCI Configuration Mechanism #1 to access the configuration spaces. Two I/O locations are used in this mechanism. The first I/O location, CF8h (which must be a double-word), references a read/write register called CONFIG_ADDRESS. The second I/O address, CFCh (which can be byte, word, or double-word), references a register called CONFIG_DATA. The general mechanism for accessing the configuration space is to write a value into CONFIG_ADDRESS that specifies the PCI bus, the device on that bus, and the configuration register in that device being accessed. A read or write to CONFIG_DATA will then cause FireStar to translate that CONFIG_ADDRESS value to the requested configuration cycle on the PCI bus. Below is an example to read PCIDV1 00h (the register located at 00h in the PCI Configuration Space of the 82C700):

```
MOV    EAX,80000800h   ;specifies the device, function,
                       and register number
MOV    DX,0CF8h        ;CONFIG_ADDRESS
OUT    DX,EAX
MOV    DX,0CFCh        ;CONFIG_DATA
IN     EAX,DX
```

The content of the CONFIG_ADDRESS shown above possesses the following meanings (device number 00001b means the 82C700 is designed to use AD12 as the IDSEL) as shown in Table 5-2.

Table 5-3 shows the correspondence of device number of the IDSEL actually generated on the PCI bus during configuration cycles.

Table 5-2 CONFIG_ADDRESS Example

31	30 24	23 16	15 11	10 8	7 2	1	0
1	Reserved	Bus Number	Device Number	Function Number	Register Number	0	0
1	000 0000	0000 0000	0000 1	000	0000 00	0	0
80h		00h	08h		00h		

Table 5-3 Device Number Decode

Device Number	IDSEL
PCIDV1 0h (82C700)	AD11
PCIDV0 1h (82C700)	AD12
2h	AD13
3h	AD14
4h	AD15
5h	AD16
6h	AD17
7h	AD18
8h	AD19
9h	AD20
Ah	AD21

Device Number	IDSEL
Bh	AD22
Ch	AD23
Dh	AD24
Eh	AD25
Fh	AD26
10h	AD27
11h	AD28
12h	AD29
13h	AD30
PCI IDE 14h (IDE Controller)	AD31

5.1 SYSCFG Register Space

An indexing scheme is used to access the System Control Register Space (SYSCFG). Port 022h is used as the Index Register and Port 024h as the Data Register. Each access to a register within this space consists of:

1) a write to Port 022h, specifying the desired register in the data byte,

2. followed by a read or write to Port 024h with the actual register data.

The index resets after every access; so every data access (via Port 024h) must be preceded by a write to Port 022h even if the same register is being accessed consecutively.

Port 023h is the Data Register for DMA clock select.

Table 5-5 gives the bit formats for the registers located at SYSCFG 00h-2Fh. Table 5-6 gives the bit formats for the registers located at SYSCFG 30h-FFh which are the power management registers.

5.1.1 System Configuration Register Index/Data Programmable

The SYSCFG index/data ports default to 022h/024h as in previous OPTi chipsets, but now these registers are accessible at other locations as well. PCIDV1 5Fh is provided to program the upper bits of the index/data port I/O address. These register bits default to 0, leaving the traditional 022h/024h locations as index/data. Refer to Table 5-4.

Table 5-4 SYSCFG Base Select Register

7	6	5	4	3	2	1	0
PCIDV1 5Fh	**Config. Register Index/ Data Port Address**						
Configuration Register Index/Data Port Address bits A[15:8]: This byte provides the upper address bits of the 16-bit address for the system configuration registers index/data port. Bits A[7:0] always point to 022h/024h. At reset this register defaults to 0, so the full I/O address for the index/data ports is 0022/0024h.							

Table 5-5 SYSCFG 00h-2Fh

7	6	5	4	3	2	1	0
SYSCFG 00h	**Byte Merge/Prefetch & Sony Cache Module Control Register**						**Default = 00h**
Enable pipelining of single CPU cycles to memory: 0 = Disable 1 = Enable	Video memory byte/word read prefetch enable: 0 = Disable 1 = Enable Setting enables/ disables the prefetching of bytes/words from PCI video memory by the CPU.	Sony SONIC-2WP support enable:(1) 0 = No Sony SONIC-2WP installed 1 = Sony SONIC-2WP installed	Byte/word merge support: 0 = Disable 1 = Enable	Byte/word merging with CPU pipelining (NA# generation) support: 0 = Disable 1 = Enable	Time-out counter for byte/word merge: 00 = 4 CPUCLKs 01 = 8 CPUCLKs 10 = 12 CPUCLKs 11 = 16 CPUCLKs Setting determines maximum time difference between two consecutive PCI byte/word writes to allow merging.		Enable internal HOLD requests to be blocked while performing byte merge: 0 = Disable 1 = Enable
(1) If bit 5 is set, ensure that the L2 cache has been disabled (i.e., set SYSCFG 02h[3:2] = 00).							
SYSCFG 01h	**DRAM Control Register 1**						**Default = 00h**
Row address HOLD after RAS# active: 0 = 2 CPUCLKs 1 = 1 CPUCLK	RAS# active/ inactive when starting a master cycle: 0 = Active (normal page mode) 1 = Inactive	RAS pulse width used during refresh: 00 = 7 CPUCLKs 01 = 6 CPUCLKs 10 = 5 CPUCLKs 11 = 4 CPUCLKs		CAS pulse width during reads: 0 = 3 CPUCLKs 1 = 2 CPUCLKs For 1 CPUCLK width, refer to SYSCFG 1Ch[0].	CAS pulse width during writes: 0 = 3 CPUCLKs 1 = 2 CPUCLKs	RAS precharge time: 00 = 6 CPUCLKs 01 = 5 CPUCLKs 10 = 4 CPUCLKs 11 = 3 CPUCLKs	

Table 5-5 SYSCFG 00h-2Fh (cont.)

7	6	5	4	3	2	1	0
SYSCFG 02h			**Cache Control Register 1**				**Default = 00h**
L2 cache size selection: **If SYSCFG 0Fh[0] = 0** 00 = 64KB 01 = 128KB 10 = 256KB 11 = 512KB	**If SYSCFG 0Fh[0] = 1** 00 = 1MB 01 = Reserved 10 = Reserved 11 = Reserved	L2 cache write policy: 00 = L2 cache write-through 01 = Adaptive writeback Mode 1 10 = Adaptive writeback Mode 2 11 = L2 cache writeback		L2 cache operating mode select: 00 = Disable 01 = Test Mode 1; External Tag Write (Tag data write-through SYSCFG 07h) 10 = Test Mode 2; External Tag Read (Tag data read from SYSCFG 07h) 11 = Enable L2 cache		DRAM posted write: 0 = Disable 1 = Enable	CAS precharge time: 0 = 2 CPUCLKs 1 = 1 CPUCLK
SYSCFG 03h			**Cache Control Register 2**				**Default = 00h**
Timing for burst writes to L2 cache: 00 = X-4-4-4 10 = X-2-2-2 01 = X-3-3-3 11 = X-1-1-1		Leadoff cycle time for writes to L2 cache: 00 = 5-X-X-X 10 = 3-X-X-X 01 = 4-X-X-X 11 = 2-X-X-X		Timing for burst reads to L2 cache: 00 = X-4-4-4 10 = X-2-2-2 01 = X-3-3-3 11 = X-1-1-1		Leadoff cycle time for reads to L2 cache: 00 = 5-X-X-X 10 = 3-X-X-X 01 = 4-X-X-X 11 = 2-X-X-X	
SYSCFG 04h			**Shadow RAM Control Register 1**				**Default = 00h**
CC000h-CFFFFh read/write control: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM		C8000h-CBFFFh read/write control: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM		Sync SRAM pipelined read cycle 1-1-1-1 enable:(1) 0 = Implies leadoff T-state for read pipe-lined cycle = 2(2) 1 = Enables leadoff T-state for read pipe-lined cycle = 1(3)	E0000h-EFFFFh range selection: Determines whether this region will be treated like the F0000 BIOS area or whether it will always be non-cacheable. 0 = E0000h-EFFFFh area will always be non-cache-able 1 = E0000h-EFFFFh area will be treated like the F0000h BIOS area. If this bit is set, then SYSCFG 06h[3:2] and [1:0] Should be set identically.	C0000h-C7FFFh read/write control: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM	

(1) If SYSCFG 03h[3:2] = 11, then this register setting is valid.
(2) It will be a 3-1-1-1 cycle followed by a 2-1-1-1 cycle, or a 3-1-1-1 cycle for successive pipelined cycles, based on SYSCFG 10h[5].
(3) It will be a 3-1-1-1 cycle followed by a 1-1-1-1 cycle for successive pipelined cycles. SYSCFG 10h[5] must be set to 1.

Table 5-5 SYSCFG 00h-2Fh (cont.)

7	6	5	4	3	2	1	0
SYSCFG 05h			**Shadow RAM Control Register 2**				**Default = 00h**
DC000h-DFFFFh read/write control: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM		D8000h-DBFFFh read/write control: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM		D4000h-D7FFFh read/write control: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM		D0000h-D3FFFh read/write control: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM	
SYSCFG 06h			**Shadow RAM Control Register 3**				**Default = 00h**
DRAM hole in system memory from 80000h-9FFFFh:(1) 0 = No hole in memory 1 = Enable hole in memory	Wait state addition for PCI master snooping: 0 = Do not add a wait state for the cycle access finish to do the snooping 1 = Add a wait state for the cycle access to finish and then do the snooping	C0000h-C7FFFh cacheability: 0 = Not cacheable 1 = Cacheable in L1 and L2 (L1 disabled by SYSCFG 08h[0])	F0000h-FFFFFh cacheability: 0 = Not cacheable 1 = Cacheable in L1 and L2 (L1 disabled by SYSCFG 08h[0])	F0000h-FFFFFh read/write control: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM If SYSCFG 04h[2] = 1, then the E0000h-EFFFFh read/write control should have the same setting as this.		E0000h-EFFFFh read/write control: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM	
(1) This setting gives the user the option to have some other device in the address range 80000h-9FFFFh instead of system memory. When bit 7 is set, the 82C700 will not start the system DRAM controller for accesses to this particular address range.							
SYSCFG 07h			**Tag Test Register**				**Default = 00h**
- Data from this register is written to the tag, if in Test Mode 1 (refer to SYSCFG 02h). - Data from the tag is read into this register, if in Test Mode 2 (refer to SYSCFG 02h).							

Table 5-5 SYSCFG 00h-2Fh (cont.)

7	6	5	4	3	2	1	0
SYSCFG 08h			**CPU Cache Control Register**				**Default = 00h**
Reserved	Snoop filtering for bus masters:(1) 0 = Disable 1 = Enable	CPU HITM# pin sample timing: 0 = Delay 1 CLK (HITM# sampled on 3rd rising edge of PCICLK after EADS# assertion) 1 = No delay (HITM# sampled on 2nd rising edge of PCICLK after EADS# assertion)	Parity checking: 0 = Disable 1 = Enable Not supported.	Reserved	CPU address pipelining for DRAM burst cycles: 0 = Disable 1 = Enable (Allow: X-2-2-2-3-2-2-2 if SYSCFG 1Fh[5] = 1 or X-2-2-2-2-2-2-2 if SYSCFG 1Fh[5] = 0 or X-2-2-2-X-2-2-2 if SYSCFG 1Fh[5] = 0 and 11[4] = 1)	L1 cache writeback and write-through control: 0 = Write-through only 1 = Writeback enabled	BIOS area cacheability in L1 cache: Determines if system BIOS area E0000h-FFFFFh (if SYSCFG 04h[2] = 1) or F0000h-FFFFFh (if SYSCFG 04h[2] = 0), and video BIOS area C0000h-C7FFFh is cacheable in L1 or not. 0 = Cacheable 1 = Not Cacheable

(1) For a master request if the subsequent read/write is within the same cache line, CPU 'Inquire' cycles are not done until there is a cache line miss (i.e., line comparator not activated for accesses within the same cache line).

7:6	5:4	3:2	1:0
SYSCFG 09h	**System Memory Function Register**		**Default = 00h**
DRAM Hole B size: 00 = 512KB 10 = 2MB 01 = 1MB 11 = 4MB Address for this hole is specified in SYSCFG 0Bh[7:0] and 0Ch[3:2]	DRAM Hole B control mode: 00 = Disable 01 = WT for L1 and L2 10 = Non-cacheable for L1 and L2 11 = Enable hole in DRAM	DRAM Hole A size: 00 = 512KB 10 = 2MB 01 = 1MB 11 = 4MB Address for this hole is specified in SYSCFG 0Ah[7:0] and 0Ch[1:0]	DRAM Hole A control mode: 00 = Disable 01 = WT for L1 and L2 10 = Non-cacheable for L1 and L2 11 = Enable hole in DRAM

SYSCFG 0Ah **DRAM Hole A Address Decode Register** **Default = 00h**

DRAM Hole A starting address:
- These bits along with SYSCFG 0Ch[1:0] are used to specify the starting address of DRAM Hole A.
- These bits, AST[7:0], map onto HA[26:19] lines.

SYSCFG 0Bh **DRAM Hole B Address Decode Register** **Default = 00h**

DRAM Hole B starting address:
- These bits along with SYSCFG 0Ch[3:2] are used to specify the starting address of DRAM Hole B.
- These bits, BST[7:0], map onto HA[26:19] lines.

Table 5-5 SYSCFG 00h-2Fh (cont.)

7	6	5	4	3	2	1	0
SYSCFG 0Ch			**DRAM Hole Higher Address**				**Default = 00h**
1 = Generate ADSC# 2 CPU clocks after CPU ADS# is asserted. This is used in case sync SRAM can't run at 3-1-1-1. Not supported.	Fast BRDY# generation for DRAM write page hits. BRDY# for DRAM writes generated on: 0 = 4th CPUCLK 1 = 3rd CPUCLK	HACALE cycle: 0 = Normal timing 1 = HACALE one-half a clock cycle early Applies to async cache. Not supported.	If set, CPU-to-PCI memory device can not be X-3-3-3 except line hit with previous cycle. This bit will have effect only if SYSCFG 1Fh[2] is turned on and CPU/PCI CLK is synchronous Not supported.	DRAM Hole B starting address: These bits are used in conjunction with the bits in SYSCFG 0Bh to specify the starting address of DRAM Hole B. These bits, BST[9:8], map onto HA[28:27].		DRAM Hole A starting address: These bits are used in conjunction with the bits in SYSCFG 0Ah to specify the starting address of DRAM Hole A. These bits, AST[9:8], map onto HA[28:27].	
SYSCFG 0Dh			**Clock Control Register**				**Default = 00h**
Sync SRAM clock source: 0 = CPU clock 1 = ECLK Not supported.	CPU memory cycle always preempts GUI: 0 = Enable 1 = Disable (UMA feature - not supported)	BOFF# generation control: 0 = No change in BOFF# generation 1 = BOFF# not generated if request from PCI-to-ISA bridge is removed before last BRDY# Recommend to set this bit if ISA refresh is enabled.	Preempt GUI whenever MGNT# is active: 0 = No change 1 = Yes (Do not set it if not necessary) (UMA feature - not supported)	Enable A0000h-BFFFFh as system memory: 0 = No 1 = Yes	Add one more wait state during PCI master cycle with Intel-type address toggling(1): 0 = No 1 = Yes	Give FireStar control of the PCI bus on STOP# generation after HITM# is active: 0 = No 1 = Yes(2)	CPU clock is slowed down to below 33MHz: 0 = No 1 = Yes

(1) If the PCI master does its address toggling in the style of the Intel 486 burst, rather than a linear burst mode style, then one wait state needs to be added.

(2) FireStar has control over the PCI bus until the writeback is completed. If PCI master pre-snoop has been enabled (SYSCFG 0Fh[7] = 1), 0Dh[1] should be set to 1.

Table 5-5 SYSCFG 00h-2Fh (cont.)

7	6	5	4	3	2	1	0
SYSCFG 0Eh			**PCI Master Burst Control Register 1**				**Default = 00h**
0 = PCI master / L2 concurrency with CPU non-burst cycle only 1 = PCI master / L2 concurrency for all cycles Not supported.	ISA/DMA master through internal MRD#/MWR#: 0 = Disable 1 = Enable This bit must be turned off for DMA support with SDRAM.	0 = Disable 1 = GUI high priority request in PCI master HITM# will not retry (UMA feature - not supported)	0 = Disable 1 = PCI master write won't be retried if GUI owns the bus (UMA feature - not supported)	Parity check during master cycles (if SYSCFG 08h[4] = 1): 0 = Enable 1 = Disable	Generate NA# for every single transfer cycle: 0 = Disable 1 = Enable	Write protection for L1 BIOS: 0 = No 1 = Yes	PCI line comparator (if SYSCFG 08h[6] = 1): 0 = Use line comparator in PCI master 1 = Generate inquire cycle for every new FRAME#
SYSCFG 0Fh			**PCI Master Burst Control Register 2**				**Default = 00h**
PCI pre-snoop: 0 = Disable 1 = Enable(1) Also see SYSCFG 0Dh[1].	Insert wait states for ISA master access: 0 = No 1 = Yes	CPU-to-DRAM deep buffer in L2 WT mode: 0 = Disable 1 = Enable	0 = Default method 1 = If internal PCICYC and BKFRAME are active, retry the PCI cycle block EADS# generation.	New mode of single cycle NA#: 0 = No change in cache write hit timing 1 = Cache write hit single transfer cycles will take 3 CLKS to comple if the line is already dirty	Generate ADSC# for sync SRAM 1 clock after CPU ADS# in read cycle: 0 = No 1 = Yes(2)	Write pulse width: 0 = 1 CPUCLK 1 = CPUCLK/2 + internal delay line Used only in 3-X-X-X write in async SRAM mode. Not supported.	Cache size selection: This bit along with SYSCFG 02h[1:0] defines the L2 cache size. 0 = < 1MB 1 = 1MB
(1) FireStar generates a pre-snoop cycle to the CPU assuming that the PCI master will do a burst. (2) SYSCFG 0Fh[2] needs to be set if pipelined sync SRAMs are being used.							
SYSCFG 10h			**Miscellaneous Control Register 1**				**Default = 00h**
CPU to PCI/ ISA slave cycle triggered: 0 = After 2nd T2 1 = After 1st T2	Cache modified write cycle timing: 0 = No delay on CA4 1 = CA4 is delayed one-half clock	Leadoff cycle for a pipelined read: 0 = 3-X-X-X read followed by a 3-X-X-X pipelined read cycle 1 = 3-X-X-X read followed by a 2-X-X-X pipelined read cycle	2-X-X-X pipelined write hit cycles: 0 = Disable 1 = Enable	Move the write pulse one-half a clock later in X-2-2-2 write hit cycles: 0 = No 1 = Yes	Move the write pulse one-half a clock earlier in 3-X-X-X write hit cycles: 0 = No 1 = Yes	Reserved	PCICLK select control:(1) 0 = PCICLK is async to CPUCLK 1 = PCICLK is sync to CPUCLK
(1) If bit 0 is set, (i.e., sync PCI implementation) then the timing constraints between the PCICLK and CPUCLK inputs to FireStar must be met. PCICLK <= CPUCLK/2 period before CPUCLK PCICLK <= 0.5ns after CPUCLK. Note that in the sync PCICLK option, PCICLK = CPUCLK/2.							

Table 5-5 SYSCFG 00h-2Fh (cont.)

7	6	5	4	3	2	1	0
SYSCFG 11h			**Miscellaneous Control Register 2**				**Default = 00h**
Reserved		Cache inactive during Idle state control: This bit controls the chip selects of the SRAMs. 0 = SRAM active always 1 = SRAM inactive during Idle state	CPU address pipelining for DRAM burst cycles: 0 = Controlled by SYSCFG 08h[2] and 1F[5] 1 = Slow pipelining (allow X-2-2-2-X-2-2-2 when SYSCFG 08h[2] = 1 and 1F[5] = 0	0 = TAG[7:0] pins as TAG[7:0] outputs 1 = TAG[7:0] pins as CAS[7:0]# outputs	Page miss posted write: 0 = Enable 1 = Disable	ATWLRDYB used to block CSX when BOFFX	Delay start: 0 = Do not delay internal master cycles after an inquire cycle 1 = Delay internal master cycles by 1 PCICLK after inquire cycle
SYSCFG 12h			**Refresh Control Register**				**Default = 00h**
REFRESH# pulse source: 0 = 82C700 or ISA master is source of REFRESH# input 1 = 32KHz clock Not supported.	Sync SRAM, linefill cache write timing: 0 = Normal 1 = Delay 1 CPUCLK	Suspend mode refresh: 00 = From CPUCLK state machine 01 = Self-refresh based on 32KHz only 10 = Normal refresh based on 32KHz only 11 = Reserved		Slow refresh: Refresh on: 00 = Every REFRESH#/32KHz falling edge 01 = Alternate REFRESH#/32KHz falling edge 10 = One in four REFRESH#/32KHz falling edge 11 = Every REFRESH#/32KHz toggle		LA[23:17] enable from 8Fh during refresh: 0 = Disable 1 = Enable	MP[7:4] output enable during PCI master write: 0 = Disable (from CPU address) 1 = Enable Not supported.
SYSCFG 13h			**Memory Decode Control Register 1**				**Default = 00h**
Reserved	Full decode for logical Bank 1 (RAS1#): 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)			SMRAM: 0 = Disable 1 = Enable See SYSCFG 14h[3]	Full decode for logical Bank 0 (RAS0#): 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)		

Table 5-5 SYSCFG 00h-2Fh (cont.)

7	6	5	4	3	2	1	0
SYSCFG 14h			**Memory Decode Control Register 2**				**Default = 00h**
Data buffer control during configuration cycles: 0 = Normal 1 = Generate internal HDOE# signal Must = 1 for EDO timing.	Full decode for logical Bank 3 (RAS3#): 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)			SMRAM control: Inactive SMIACT#: 0 = Disable SMRAM 1 = Enable SMRAM(1) Active SMIACT#: 0 = Enable SMRAM for both Code and Data(1) 1 = Enable SMRAM for Code only(1)	Full decode for logical Bank 2 (RAS2#): 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)		
(1) If SYSCFG 13h[3] is set.							
SYSCFG 15h			**PCI Cycle Control Register 1**				**Default = 00h**
CPU master to PCI memory slave write IRDY# control: 00 = 3 PCICLKs after data 01 = 2 PCICLKs after data 10 = 1 PCICLK after data 11 = 0 PCICLK after data		CPU master to PCI slave write posting, bursting control: 00 = No posting, no bursting 01 = Posting only, no bursting 10 = Posting, with conservative bursting 11 = Posting, with aggressive bursting		Master retry timer: Selects the delay before retry is attempted. 00 = 10 PCICLKs 01 = 18 PCICLKs 10 = 34 PCICLKs 11 = 66 PCICLKs		Reserved	PCI FRAME# generation control: 0 = Conservative mode in CPU pipelined cycle 1 = Aggressive mode
SYSCFG 16h			**Dirty/Tag RAM Control Register**				**Default = A0h**
This bit along with bit 5 and PCICLK3 strap define DIRTY, CMD# as PCICLK3 on the CMD# pin, and CACS# or DIRTY options on the CACS# pin.(1)	Reserved	Tag RAM size selection: 0 = 8-bit 1 = 7-bit (Default) Selects CACS# for 7-bit and DIRTY for 8-bit tag(1)	Single write hit leadoff cycle in a combined Dirty/Tag implementation 0 = 5 cycles 1 = 4 cycles	Pre-snoop control: 0 = Pre-snoop for starting address 0 only 1 = Pre-snoop for all addresses except those on the line boundary	Synchronization between PCICLK and CPUCLK: 0 = PCICLK async to CPUCLK 1 = PCICLK sync to CPUCLK (skew not to exceed –2ns to 15ns)	Reserved	Internal HDOE# timing control: 0 = Negated normally 1 = Negated one clock before the cycle finishes

(1) **ROMCS#:KBDCS# strapped for CMD#:**

Bits 7 & 5	CACS# Pin	CMD# Pin
00	CACS#	DIRTY
01	CACS#	DIRTY
10	DIRTY	CMD#
11	CACS#	CMD#

(1) **ROMCS#:KBDCS# strapped for PCICLK3:**

Bits 7 & 5	CACS# Pin	CMD# Pin
00	DIRTY	PCICLK3
01	CACS#	PCICLK3
10	DIRTY	PCICLK3
11	CACS#	PCICLK3

Table 5-5 SYSCFG 00h-2Fh (cont.)

7	6	5	4	3	2	1	0
SYSCFG 17h			**PCI Cycle Control Register 2**				**Default = 00h**
Reserved	Generate NA# for PCI slave access in async PCICLK mode: 0 = No 1 = Yes	Two banks of sync SRAM are installed: 0 = Disable 1 = Enable Not supported.	Reserved		Pipelining during byte merge: 0 = Disable 1 = Enable	Sync SRAM type (if SYSCFG 11h[3] = 1): 0 = Standard 1 = Pipelined	Burst type: 0 = Intel burst protocol 1 = Cyrix linear burst protocol
SYSCFG 18h			**Interface Control Register**				**Default = 00h**
Reserved	Drive strength on RAS lines: 0 = 16mA 1 = 4mA	CAS lines voltage selection: 0 = 5.0V 1 = 3.3V	Drive strength on memory address lines and write enable line: 0 = 4mA 1 = 16mA	Reserved			
SYSCFG 18h - FS ACPI Version			**Interface Control Register**				**Default = 00h**
Drive strength on SDRAS and SDCAS lines: 0 = 16mA 1 = 4mA	Drive strength on RAS lines: 0 = 16mA 1 = 4mA	CAS lines voltage selection: 0 = 5.0V 1 = 3.3V	Drive strength on memory address lines: 0 = 4mA 1 = 16mA	Drive strength on write enable line: 0 = 16mA 1 = 20mA	Reserved	Reserved	Test bit: 0 = Default method 1 = No latch for TAG, use as internal test mode. For test only, do not use. Write to 0.
SYSCFG 19h			**Memory Decode Control Register 3**				**Default = 00h**
Pin functionality: 0 = GWE# 1 = RAS5#	Full decode for logical Bank 5 (RAS5#) if SYSCFG 19h[7] is set: 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)			Bank 4 (RAS4#): 0 = Disable 1 = Enable	Full decode for logical Bank 4 (RAS4#): 000 = 0Kx36 001 = 256Kx36 (2MB) 010 = 512Kx36 (4MB) 011 = 1Mx36 (8MB) 100 = 2Mx36 (16MB) 101 = 4Mx36 (32MB) 110 = 8Mx36 (64MB) 111 = 16Mx36 (128MB)		
SYSCFG 1Ah			**Memory Shadow Control Register 1**				**Default = 00h**
SLIC: 0 = Disable 1 = Enable	Time that CPU is ensured for bus utilization during every 15µs of system operation:(1) 00 = No bandwidth guarantee 01 = 1µs guarantee 10 = 2µs guarantee 11 = 4µs guarantee		C8000h-DFFFFFh shadowing granularity: 0 = 16KB 1 = 8KB	Read and write control of CE000h-CFFFFh for shadowing if SYSCFG 1Ah[4] = 1: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM		Read and write control of CA000h-CBFFFh for shadowing if SYSCFG 1Ah[4] = 1: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM	

(1) Bits [6:5] allow the user to guarantee the CPU a percentage of the total available bus bandwidth. When these bits are programmed, the CPU is ensured of utilization of the bus for up to 4µs of every 15µs of operation of the system. This is achieved by not granting bus ownership to other requesting devices.

Table 5-5 SYSCFG 00h-2Fh (cont.)

7	6	5	4	3	2	1	0
SYSCFG 1Bh			**Memory Shadow Control Register 2**				**Default = 00h**
Read and write control of DE000h-DFFFFh for shadowing if SYSCFG 1Ah[4] = 1: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM		Read and write control of DA000h-DBFFFh for shadowing if SYSCFG 1Ah[4] = 1: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM		Read and write control of D6000h-D7FFFh for shadowing if SYSCFG 1Ah[4] = 1: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM		Read and write control of D2000h-D3FFFh for shadowing if SYSCFG 1Ah[4] = 1: 00 = Read/write PCI bus 01 = Read from DRAM/write to PCI 10 = Read from PCI/write to DRAM 11 = Read/write DRAM	
SYSCFG 1Ch			**EDO DRAM Control Register**				**Default = 00h**
Bank 5: 0 = FPM DRAM 1 = EDO DRAM	Bank 4: 0 = FPM DRAM 1 = EDO DRAM	Bank 3: 0 = FPM DRAM 1 = EDO DRAM	Bank 2: 0 = FPM DRAM 1 = EDO DRAM	Bank 1: 0 = FPM DRAM 1 = EDO DRAM	Bank 0: 0 = FPM DRAM 1 = EDO DRAM	82C700 operating at a frequency of 50MHz:[1] 0 = No 1 = Yes Also see SYSCFG 1Dh[7].	CAS pulse width during DRAM accesses: 0 = CAS pulse width determined by SYSCFG 01h[3] 1 = CAS pulse width is 1 CPUCLK[2]
(1) Bit 1 can be set by the BIOS when FireStar is operating at <= 50MHz. The setting of this bit can improve DRAM access times by allowing X-2-2-2 burst to DRAM even if the user is not using EDO DRAMs, but uses 60ns fast page mode DRAM instead. (2) The width of the pulse is one CPUCLK for read accesses to banks that are populated with EDO DRAMs (selected by bits [7:2]), resulting in X-2-2-2 burst to EDO DRAM at 50/60/66MHz. SYSCFG 14h[7] and PCIDV0 44h[0] must be set in prior to setting this bit. X-2-2-2 burst cycles enabled by this bit apply only during CPU read bursts to EDO DRAM banks that are enabled in SYSCFG 1Ch[7:2].							
SYSCFG 1Dh			**Miscellaneous Control Register 3**				**Default = 00h**
Generate internal HDOE# signal one-half clock earlier during CPU reads from DRAM: 0 = Disable 1 = Enable Only for 50MHz with X-2-2-2 operation.	When set (to 1), address latching in DRAM module will take place on the 3rd clock after ADS# if fast write posting is enabled through SYSCFG 0Ch[6] and CPU-to-DRAM buffer is not enabled. Not supported.	DWE# timing selection:[1] 0 = Normal 1 = Removed 1 CPUCLK earlier	DRAM read leadoff cycle: 0 = Normal 1 = Reduced by 1 CPUCLK	DMA accesses from system memory: 0 = Enable 1 = Disable	When set (to 1) PEN becomes MPERR# input. Not supported.	Accesses to B0000h-BFFFFh during SMM mode: 0 = Accesses go to main memory 1 = Accesses go to PCI bus	Accesses to A0000h-AFFFFh during SMM mode: 0 = Accesses go to main memory 1 = Accesses go to PCI bus
(1) When using a buffered DWE# solution and the DRAM load is substantial, bit 5 may have to be set if the system begins to malfunction.							

Table 5-5 SYSCFG 00h-2Fh (cont.)

7	6	5	4	3	2	1	0
SYSCFG 1Eh			**Control Register**				**Default = 00h**
PCI master read cycle: 0 = Wait for IRDY# to be asserted before asserting TRDY# 1 = Generate TRDY# without checking for the status of IRDY#	GUI block FRAME# one CLK earlier: 0 = Disable 1 = Enable (UMA feature - not supported)	Retry PCI pre-snoop HITM# cycle: 0 = Disable 1 = Enable	BOFF# generation if the PCI retry cycle is in 80000h-FFFFFh range: 0 = Not generated 1 = Generated **Note:** Bit 3 must = 1, otherwise the setting of this bit has no effect.	Deadlock situation:(1) 0 = No way to avert deadlock situation if write posting buffer on the PCI-to-PCI bridge has been enabled 1 = BOFF# is asserted to the CPU if deadlock situation occurs	Must be set to 1 to correct glitch on DWE# output pin. This bit works in conjunction with SYSCFG 20h[7].	When set to 1, PCI bursting will be disabled if BE[7:4]# and/or BE[3:0]# are not all 0.	Reserved
(1) In a situation where there is a PCI-to-PCI bridge in a system and that bridge supports write posting, the following deadlock condition can occur. The bridge posts data from a master on the secondary PCI bus into its FIFO. If at the same time the 82C700 is accessing the bridge as a target, then the bridge will tell the 82C700 to retry its request after it has serviced out its FIFO. This will result in a deadlock situation. Bit 3 needs to be set to 1 if an OPTi 82C824 or 82C814 bridge, or a DEC 21050 PCI-to-PCI bridge (or a similar chip) is used.							
SYSCFG 1Fh			**EDO Timing Control Register**				**Default = 00h**
0 = Normal 1 = Generate conflict during EDO detection (bit 6 set) if necessary	0 = Normal (fast page mode) 1 = Detect EDO	NA# generation for burst DRAM accesses: 0 = Aggressive (X-2-2-2-2-2-2-2 if SYSCFG 08h[2] = 1) 1 = Controlled by SYSCFG 08h[2] Also see SYSCFG 11h[4]	DRAM read cycle leadoff reduced by 1 clock to support 5-2-2-2 at 50MHz: 0 = No (normal) 1 = Yes	Reserved	0 = Async SRAM use 8 CS# and one WE#. 1 = Async SRAM use one CS# and 8 WE#, swap CS# and ECAWE# in this mode. This is only good for single bank cache. Also ECA4 and OCA4 are swapped. Not supported.	PCI write triggering: 0 = Normal 1 = Default Method	0D0000-0DFFFFh is cacheable in L1 and L2:(1) 0 = No 1 = Yes
(1) Before turning on bit 0, 0D0000-0DFFFFh needs to be readable/writable and shadowed. When cached into L1, it will be in writeback mode if SYSCFG 08h[1] = 1. There is no write protection in this region if bit 0 is set.							

Table 5-5 SYSCFG 00h-2Fh (cont.)

7	6	5	4	3	2	1	0
SYSCFG 20h			**DRAM Burst Control Register**				**Default = 00h**
Must be set to 1 to correct glitch on DWE# output pin. This bit works in conjunction with SYSCFG 1Eh[2].	DRAM post write during HITM# cycle during PCI master access: 0 = Disable 1 = Enable	0 = IRDY# inactive for > 3 clocks 1 = Lockup may occur Must be set to 0 to correct lockup if a master IRDY# is not asserted within 3 CLKs after FRAME#.	PCI master parity: 0 = Disable 1 = Enable	DRAM write burst cycle control during PCI master cycles: 00 = Invalid 01 = X-3-3-3 10 = X-2-2-2 11 = X-1-1-1		DRAM read burst cycle control during PCI master cycles: 00 = Invalid 01 = X-3-3-3 10 = X-2-2-2 11 = X-1-1-1	
SYSCFG 21h			**PCI Concurrency Control Register**				**Default = 01h**
Concurrency timer: 0 = Conservative 1 = Aggressive	00 = No concurrency on PCI master and CPU/L2 If SYSCFG 21h[1] = 1, then: X1 = PCI master and CPU/L2 concurrence for PCI write invalid cycles 1X = PCI master and CPU/L2 concurrence for PCI read multiple and read line cycles		Concurrency on PCI master-PCI slave, and CPU/L2/DRAM: 0 = No 1 = Yes	0 = Normal Tag write 1 = If bit 1 is set, always write invalid Tag during linefill	0 = If Tag = 11011111b => invalid combination 1 = If cache size = 256K, Tag = 0000 1100b => invalid combination (CF0000h). If cache size > 256K, Tag = 10111111b => invalid combination Valid only when bit 1 = 1.	L2 cache write mode during master cycle: 0 = Write-through 1 = Writeback	0 = HOLD/HLDA protocol 1 = BOFF#/AHOLD protocol (Default) Must be set to 1. HOLD/HLDA protocol not supported on FireStar.
SYSCFG 22h			**Inquire Cycle Control Register**				**Default = 00h**
HLDA inactive: 0 = Yes 1 = No Must be set to 0. HOLD/HLDA protocol not supported on FireStar.	Reserved	Must be set to 0 to correct glitch on (internal) HRQ signal.	HRQ is sync to PCICLK: 0 = No 1 = Yes Must = 1 for DDMA operation	0 = No write allocation 1 = CPU single write, L2 miss cache allocation	EADS# generation: 00 = Normal for inquire cycle 01 = 1 CPUCLK earlier 10 = 1 CPUCLK earlier with async clock, 2 CPUCLK earlier with sync clock 11 = Reserved		Inquire cycle to PCI clock synchronization: 0 = Use rising edge only (old mode) 1 = Use rising and falling edges

Table 5-5 SYSCFG 00h-2Fh (cont.)

7	6	5	4	3	2	1	0
SYSCFG 23h			**Pre-Snoop Control Register**				**Default = 00h**
Generate internal BREAK signal during master accessing of local memory cycle:[(1)] 0 = Default Mode 1 = New Mode	0 = Bank 0 is the first bank 1 = Bank 0 becomes the last bank and Bank 1 is the first bank Purpose of this bit is to give system a choice when UMA is supported. (UMA feature - not supported)	Pre-snoop for PCI X-1-1-1 write invalidate: 0 = Disable 1 = Enable	Pre-snoop for PCI X-1-1-1 read multiple and read line: 0 = Disable 1 = Enable	Half clock shift of cache hit latching when fast NA is enabled: 0 = Disable 1 = Enable	Fix a glitch on GUIP: 0 = No glitch removal 1 = Remove glitch on GUIP signal (UMA feature - not supported)	0 = Normal 1 = MREQ# block UMA feature - not supported)	Reserved
(1) Default Mode conditions: Sync SRAM, starting address AD[4:2] not = 000 or non-linear mode, master L2 cache write-through. New Mode conditions: Sync SRAM, starting address AD[4:2] not = 000 or non-linear mode, master L2 cache write-through, L2 cache hit.							
SYSCFG 24h			**Asymmetric DRAM Configuration Register**				**Default = 00h**
Logical Bank 3 DRAM type: 00 = Sym DRAM 01 = Asym DRAM - x8 type 10 = Asym DRAM - x9 type 11= Asym DRAM - x10 type		Logical Bank 2 DRAM type: 00 = Sym DRAM 01 = Asym DRAM - x8 type 10 = Asym DRAM - x9 type 11= Asym DRAM - x10 type		Logical Bank 1 DRAM type: 00 = Sym DRAM 01 = Asym DRAM - x8 type 10 = Asym DRAM - x9 type 11= Asym DRAM - x10 type		Logical Bank 0 DRAM type: 00 = Sym DRAM 01 = Asym DRAM - x8 type 10 = Asym DRAM - x9 type 11= Asym DRAM - x10 type	
SYSCFG 25h			**GUI Memory Location Register**				**Default = 00h**
GUI memory location: A[31:27] (UMA feature - not supported)					UMA size: 0 = Decided by SYSCFG 26h[5:4] 1 = 0.5MB if SYSCFG 26h[5:4] = 00 (UMA feature - not supported)	Reserved	Split buffer present: 0 = No 1 = Yes UMA feature - not supported)

Table 5-5 SYSCFG 00h-2Fh (cont.)

7	6	5	4	3	2	1	0
SYSCFG 26h			**UMA Control Register 1**				**Default = 00h**
ISA master to DRAM cycle CAS width: 0 = Controlled by ISA R/W command pulse width 1 = 2 PCICLKs This bit is effective only when SYSCFG 0Eh[6] = 1	ISA SA address latch: 0 = SA latch is always transparent (pass-through) 1 = SA latch is on for retry only. (When first CPU/ISA cycle is retried, SA address will be latched.)	GUI memory size: 00 = 1MB 01 = 2MB 10 = 3MB 11= 4MB For 0.5MB size, set these bits to 00 and SYSCFG 25h[2] = 1. (UMA feature - not supported)		5-2-2-2 EDO DRAM read timing at 66MHz in a cacheless system: 0 = Disable 1 = Enable	00 = Normal 01 = For low priority GUI request, 82C700 will wait for two more CPUCLKs 10 = Reserved 11 = GUI is always at high priority (UMA feature - not supported)		UMA support: 0 = Disable 1 = Enable (UMA feature - not supported)
SYSCFG 27h			**Miscellaneous Control Register 4**				**Default = 00h**
Master to EDO DRAM read cycle controlled by DWE#: 0 = Disable 1 = Enable	Dynamic cache write hit lead-off 3 clock: 0 = Disable 1 = Enable, only valid for 4-X-X-X write hit	PCI master write line invalid cycle HITM# or L2 dirty no stopping: 0 = Disable 1 = Enable	Generate AHOLD at 2nd T2 on CPU single write hit not Dirty cycle: 0 = Disable 1 = Enable	Fast NA# with L2 cache: 0 = Disable 1 = Enable	Non-ISA refresh counter: 000 = Disable, use external refresh pin 001 = Reserved 010 = Reserved 011 = Reserved 100 = 66MHz external CPU clock 101 = 60MHz external CPU clock 110 = 50MHz external CPU clock 111 = 40MHz external CPU clock		
SYSCFG 28h			**SDRAM Control Register 1**				**Default = 00h**
Delay CS#: 0 = Disable 1 = Enable	SDRAM CAS# latency: 001 = 1 010 = 2 011 = 3 All other combinations = Reserved			Write-through: 0 = Sequential 1 = Interleaved	SDRAM burst length control: 000 = 1 010 = 4 All other combinations = Reserved		
SYSCFG 29h			**SDRAM Control Register 2**				**Default = 00h**
Pipeline read: 0 = 7-1-1-1-5-1-1-1-1 1 = 7-1-1-1-2-1-1-1-1	2N rule: 0 = Disable 1 = Enable	Timing control: **tRP tRAS tMRS** 00 = 2 CLK 4 CLK 3 CLK 01 = 4 CLK 5 CLK 3 CLK 10 = 3 CLK 6 CLK 2 CLK 11 = Rsvd 7 CLK Rsvd tRP: Precharge time to activate command tRAS: RAS active to precharge command time tMRS: Mode register set cycle time		Bank 3 SDRAM: 0 = Disable 1 = Enable	Bank 2 SDRAM: 0 = Disable 1 = Enable	Bank 1 SDRAM: 0 = Disable 1 = Enable	Bank 0 SDRAM: 0 = Disable 1 = Enable

Table 5-5 SYSCFG 00h-2Fh (cont.)

7	6	5	4	3	2	1	0
SYSCFG 2Ah			**PCI-to-DRAM Control Register 1**				**Default = 00h**
Back-to-back PCI cycle control: 0 = Enable PCI master cycle to be pending 1 = All PCI master cycles will be retried if the previous cycle doesn't finish Not supported.	Time-out selection during PCI master read cycles when a GUI request is made: 00 = FP mode, grant the DRAM bus to GUI ASAP 01 = Select only either SDRAM or EDO time-out depending on current bank information 1X = Selects either FP mode, SDRAM, or EDO depending on current bank information (UMA feature - not supported)		PCI TRDY# wait state control with PCI-to-DRAM deep buffer: 0 = 0WS (X-1-1-1) 1 = 1WS (X-2-2-2)	Write burst with PCI-to-DRAM deep buffer: 0 = Disable 1 = Enable	Read burst with PCI-to-DRAM deep buffer: 0 = Disable 1 = Enable	PCI-to-DRAM deep buffer size: 0 = Determined by bit 0 1 = 32 dword, overrides bit 0 Not supported.	PCI-to-DRAM deep buffer size: 0 = 16 dword 1 = 24 dword
SYSCFG 2Bh			**PCI-to-DRAM Control Register 2**				**Default = 00h**
SDRAM time-out count during a GUI request: The register value plus 9 is the number of CPUCLKs delaying the GUI request to stop the DRAM controller. (UMA feature - not supported)				EDO time-out count during a GUI request: The register value plus 6 is the number of CPUCLKs delaying the GUI request to stop the DRAM controller. (UMA feature - not supported)			
SYSCFG 2Ch			**CPU-to-DRAM Buffer Control Register**				**Default = 00h**
CPU-to-PCI read and CPU-to-DRAM write concurrency: 0 = Disable 1 = Enable	This bit needs SYSCFG 2Ch[5] to be enabled. When set (to 1), the cache write to CPU-to-DRAM buffer becomes more aggressive. Will save approximately 3 clocks over the previous method. Not supported.	When set (to 1) along with CPU-to-DRAM buffer and PBSRAM, the DRAM controller will first supply the data to the CPU before writing the previous data back to DRAM during a cache miss dirty cycle.(1)	CPU-to-PCI write and CPU-to-DRAM read concurrency: 0 = Disable 1 = Enable	Enable internal LMEM# during special cycles: 0 = No 1 = Yes	BOFF# assertion during DRAM read cycles: 0 = Disable 1 = Enable	Data merging when CPU owns DRAM bus: 0 = Possible only when GUI owns DRAM bus (UMA feature - not supported) 1 = Always possible	Allow data collection while CPU-to-DRAM FIFO is flushing: 0 = Disable(2) 1 = Enable
(1) Bit 5's function needs the CPU-to-DRAM buffer to be enabled. DRAM processing for the read will start concurrently while data from cache will be written to the CPU-to-DRAM buffer. (2) BOFF# is generated for the next DRAM write cycle as long as there is data in the FIFO.							
SYSCFG 2Dh			**Miscellaneous Control Register 5**				**Default = 00h**
Split buffer concurrency: 0 = Disable 1 = Enable(1)	Predictive reading: 0 = Disable 1 = Enable	Bankwise selection for 5-X-X-X at 66MHz or 4-X-X-X at 50MHz DRAM read cycle: 0 = Default setting 1 = 5-X-X-X/4-X-X-X					
(1) Even if the CPU-to-DRAM buffer is not empty, a read/write to a non-shared memory space can continue when the GUI owns the memory bus. (UMA feature - not supported)							

Table 5-5 SYSCFG 00h-2Fh (cont.)

7	6	5	4	3	2	1	0
SYSCFG 2Eh			**UMA Control Register 2**				**Default = 00h**
Allow SDRAM self-refresh in Suspend mode: 0 = Disable (SDRAM engages auto-refresh mode) 1 = Enable (need to enable SDRAM self-refresh if SYSCFG 12h[5:4] = 01 or 10)	Allow RFSH# signal from IPC to connect to DRAM controller: 0 = Disable 1 = Enable	Allow SDRAM on RAS4#: 0 = Disable 1 = Enable Not supported.	0 = Extra half clock CPU hold time for pipeline cycle (Default) 1 = No hold time for pipeline cycle	CPU-to-PCI FIFO control module: 0 = Disable 1 = Enable	Number of address posting: 0 = 6 level 1 = 3 Level	PCI master HITM# cycle if GUI high priority request jumps in before first BRDY#: 0 = Retry all PCI cycles 1 = Retry only PCI master read (UMA feature - not supported)	PCI master request retries during GUI cycles: 0 = All PCI master requests retried 1 = PCI master read retried, write accepted (UMA feature - not supported)

Table 5-5 SYSCFG 00h-2Fh (cont.)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">SYSCFG 2Fh — UMA Control Register 3 — Default = 00h</td></tr>
<tr><td>Column address to CAS delay <u>for page miss cycles:</u>
0 = Default
1 = 1 CLK</td><td><u>See Below</u></td><td><u>0 = Only at Idle & Run state GUI will be granted even though it is a non-share bank cycle.</u>
<u>1 = GUI will be granted immediately when it is a non-share bank cycle</u>
<u>(UMA feature - not supported)</u></td><td colspan="2"><u>See Below</u></td><td colspan="3">Refresh enabling and refresh ahead control:
<u>000 = Burst refresh disable</u>
<u>001 = Always start with Bank 0, no refresh ahead</u>
<u>010 = Always start with Bank 0, refresh ahead up to 3</u>
<u>011 = Always start with Bank 0, refresh ahead up to 7</u>
<u>100 = Burst refresh disable</u>
<u>101 = Dynamic starting bank, no refresh ahead</u>
<u>110 = Dynamic starting bank, refresh ahead up to 3</u>
<u>111 = Dynamic starting bank, refresh ahead up to 7</u>

Note: This feature should never be enabled if SDRAM is being used in the system.</td></tr>
<tr><td colspan="4">Bits 6, 4, and 3: Burst mode and length selection
000 = Mode 0, RWM = 5
001 = Mode 1, RWM = 5, BLEN = 2
010 = BLEN = 3
011 = BLEN = 4
100 = Mode 0, RWM = 4
101 = Mode 2, RWM = 4, BLEN = 1
110 = BLEN = 2
111 = BLEN = 3</td><td colspan="4">RWM: Refresh request water mark.
BLEN: Minimum number of refresh cycles in a burst.
Mode 0: Refresh request is generated at reaching/crossing RWM. Refresh burst is preempted, at the end of current cycle, if high priority GUI request is pending. Refresh burst is preempted once the count drops below RWM if CPU/PCI request is pending. Otherwise the refresh continues until count is zero and then refreshes ahead up to 3/7 refreshes.
Mode 1: Refresh request is generated at reaching/crossing RWM. Refresh burst is preempted, at the end of current cycle, if high priority GUI request is pending. Refresh burst is preempted, once the count drops below RWM and number of refresh cycle performed is greater or equal to the BLEN, if CPU/PCI request is pending. Otherwise the refresh continues until count is zero and then refreshes ahead up to 3/7 refreshes.
Mode 2: Refresh request is generated at reaching/crossing RWM. Refresh burst is preempted, at the end of current cycle, if high priority GUI request is pending. Refresh burst is preempted, once number of refresh cycle performed is greater or equal to the BLEN, if CPU request is pending. refresh burst is preempted, once the count drops below RWM and number of refresh cycle performed is greater or equal to the BLEN, if PCI request is pending. Otherwise the refresh continues until count is zero and then refreshes ahead up to 3/7 refreshes.</td></tr>
</table>

Table 5-6 SYSCFG 30h-FFh (Power Management)

7	6	5	4	3	2	1	0
SYSCFG 30h-37h			**Reserved**				**Default = 00h**
SYSCFG 38h			**NMI Trap Enable Register 1**				**Default = 00h**
PMI#7 NMI: 0 = Disable 1 = Enable	PMI#6 NMI: 0 = Disable 1 = Enable	PMI#5 NMI: 0 = Disable 1 = Enable	PMI#4 NMI: 0 = Disable 1 = Enable	PMI#3 NMI: 0 = Disable 1 = Enable	PMI#2 NMI: 0 = Disable 1 = Enable	PMI#1 NMI: 0 = Disable 1 = Enable	PMI#0 NMI: 0 = Disable 1 = Enable
SYSCFG 39h			**NMI Trap Enable Register 2**				**Default = 00h**
PMI#15 NMI: 0 = Disable 1 = Enable	PMI#14 NMI: 0 = Disable 1 = Enable	PMI#13 NMI: 0 = Disable 1 = Enable	PMI#12 NMI: 0 = Disable 1 = Enable	PMI#11 NMI: 0 = Disable 1 = Enable	PMI#10 NMI: 0 = Disable 1 = Enable	PMI#9 NMI: 0 = Disable 1 = Enable	PMI#8 NMI: 0 = Disable 1 = Enable
SYSCFG 3Ah			**NMI Trap Enable Register 3**				**Default = 00h**
PMI#23 NMI: 0 = Disable 1 = Enable	PMI#22 NMI: 0 = Disable 1 = Enable	PMI#21 NMI: 0 = Disable 1 = Enable	PMI#20 NMI: 0 = Disable 1 = Enable	PMI#19 NMI: 0 = Disable 1 = Enable	PMI#18 NMI: 0 = Disable 1 = Enable	PMI#17 NMI: 0 = Disable 1 = Enable	PMI#16 NMI: 0 = Disable 1 = Enable
SYSCFG 3Bh			**NMI Trap Enable Register 4**				**Default = 00h**
PMI#31 NMI: 0 = Disable 1 = Enable	PMI#30 NMI: 0 = Disable 1 = Enable	PMI#29 NMI: 0 = Disable 1 = Enable	PMI#28 NMI: 0 = Disable 1 = Enable	PMI#27 NMI: 0 = Disable 1 = Enable	PMI#26 NMI: 0 = Disable 1 = Enable	PMI#25 NMI: 0 = Disable 1 = Enable	PMI#24 NMI: 0 = Disable 1 = Enable
SYSCFG 3Ch			**NMI Trap Enable Register 5**				**Default = 00h**
Reserved		PMI#37 NMI: 0 = Disable 1 = Enable	PMI#36 NMI: 0 = Disable 1 = Enable	PMI#35 NMI: 0 = Disable 1 = Enable	PMI#34 NMI: 0 = Disable 1 = Enable	PMI#33 NMI: 0 = Disable 1 = Enable	PMI#32 NMI: 0 = Disable 1 = Enable
SYSCFG 3Dh-3Fh			**Reserved**				**Default = 00h**
SYSCFG 40h			**PMU Control Register 1**				**Default = 00h**
Test bit for counters using 32KHz: 0 = Test Disable 1 = Test Enable For test only, do not use.	Global timer divide: 0 = ÷1 1 = ÷4	LLOWBAT polarity: 0 = Active high 1 = Active low	LOWBAT polarity: 0 = Active high 1 = Active low	Reserved	EPMI1# polarity: 0 = Active high 1 = Active low	EPMI0# polarity: 0 = Active high 1 = Active low	RSMRST select: 0 = Disable 1 = Enable Allows RESET# and RSTDRV to be generated in Resume.

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG 41h			**DOZE_TIMER Register**				**Default = 00h**
DOZE_0 time-out select: 000 = 2ms 001 = 4ms 010 = 8ms 011 = 32ms 100 = 128ms 101 = 512ms 110 = 2s 111 = 8s Time-out generates PMI#27.			Doze mode STPCLK# modulation (STPCLK# modulated by BCLK defined in SYSCFG E6h[7:6]): 000 = No Modulation (STPCLK# = 1) 001 = STPCLK# t_{hi} = 0.75 * 16 BCLKs 010 = STPCLK# t_{hi} = 0.5 * 16 BCLKs 011 = STPCLK# t_{hi} = 0.25 * 16 BCLKs 100 = STPCLK# t_{hi} = 0.125 * 16 BCLKs 101 = STPCLK# t_{hi} = 0.0625 * 16 BCLKs 110 = STPCLK# t_{hi} = 0.03125 * 32 BCLKs 111 = STPCLK# t_{hi} = 0.015625 * 64 BCLKs			ACCESS events reset Doze mode: 0 = Disable 1 = Enable	Doze control select: 0 = Hardware 1 = Software
SYSCFG 42h if AEh[7] = 0			**Clock Source Register 1**				**Default = 00h**
Clock source for GNR1_TIMER		Clock source for KBD_TIMER		Clock source for DSK_TIMER		Clock source for LCD_TIMER	
SYSCFG 42h if AEh[7] = 1			**Clock Source Register 1A**				**Default = 00h**
Clock source for GNR5_TIMER		Reserved					
SYSCFG 43h			**PMU Control Register 2**				**Default = 00h**
LCD_ACCESS includes I/O range 3B0h-3DFh: 0 = Yes 1 = No	LCD_ACCESS includes memory A0000-BFFFFh: 0 = Yes 1 = No	LOWBAT pin sample rate: 00 = 32s 10 = 128s 01 = 64s 11 = Rsrvd A PMI is generated each time LOWBAT is sampled active.		Reserved			
SYSCFG 44h			**LCD_TIMER Register**				**Default = 00h**
Time count byte for LCD_TIMER: Monitors LCD_ACCESS. Time-out generates PMI#8.							
SYSCFG 45h			**DSK_TIMER Register**				**Default = 00h**
Time count byte for DSK_TIMER: Monitors DSK_ACCESS. Time-out generates PMI#9.							
SYSCFG 46h			**KBD_TIMER Register**				**Default = 00h**
Time count byte for KBD_TIMER: Monitors KBD_ACCESS. Time-out generates PMI#10.							
SYSCFG 47h if AEh[7] = 0			**GNR1_TIMER Register**				**Default = 00h**
Time count byte for GNR1_TIMER: Monitors GNR1_ACCESS. Time-out generates PMI#11.							
SYSCFG 47h if AEh[7] = 1			**GNR5_TIMER Register**				**Default = 00h**
Time count byte for GNR5_TIMER: Monitors GNR5_ACCESS. Time-out generates PMI#11.							
SYSCFG 48h if AEh[7] = 0			**GNR1 Base Address Register**				**Default = 00h**
GNR1_ACCESS base address: A[8:1] (I/O) or A[22:15] (Memory)							
SYSCFG 48h if AEh[7] = 1			**GNR5_Timer Base Address Register**				**Default = 00h**
GNR5_TIMER base address: A[8:1] (I/O)							

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG 49h if AEh[7] = 0			**GNR1 Control Register**				**Default = 00h**
GNR1 base address: A9 (I/O) A23 (Memory)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	GNR1 mask bits for address A[5:1] (I/O) or A[19:15] memory: A 1 in a particular bit means that the corresponding bit at SYSCFG 48h[4:0] is not compared. This is used to determine address block size.				
SYSCFG 49h if AEh[7] = 1			**GNR5_Timer Control Register**				**Default = 00h**
Base address: A9 (I/O)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	GNR5 base address A[5:1] (I/O)				
SYSCFG 4Ah			**Chip Select 0 Base Address Register**				**Default = 00h**
GPCS0# base address: A[8:1] (I/O) or A[22:15] (Memory)							
SYSCFG 4Bh			**Chip Select 0 Control Register**				**Default = 00h**
GPCS0# base address: A9 (I/O) A23 (Memory)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	Chip select active: 0 = w/Cmd 1 = Before ALE	GPCS0# mask bits for address A[4:1] (I/O) or A[18:15] memory: A 1 in a particular bit means that the corresponding bit at SYSCFG 4Ah[3:0] is not compared. This is used to determine address block size.			
SYSCFG 4Ch			**Chip Select 1 Base Address Register**				**Default = 00h**
GPCS1# base address: A[8:1] (I/O) or A[22:15] (Memory)							
SYSCFG 4Dh			**Chip Select 1 Control Register**				**Default = 00h**
GPCS1# base address: A9 (I/O) A23 (Memory)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	Chip select active: 0 = w/Cmd 1 = before ALE	GPCS1# mask bits for address A[4:1] (I/O) or A[18:15] memory: A 1 in a particular bit means that the corresponding bit at SYSCFG 4Ch[3:0] is not compared. This is used to determine address block size.			
SYSCFG 4Eh if AEh[7] = 0			**Idle Reload Event Enable Register 1**				**Default = 00h**
GPCS1#_ ACCESS: 0 = Disable 1 = Enable	GPCS0#_ ACCESS: 0 = Disable 1 = Enable	LPT_ ACCESS: 0 = Disable 1 = Enable	GNR3_ ACCESS: 0 = Disable 1 = Enable	GNR1_ ACCESS: 0 = Disable 1 = Enable	KBD_ ACCESS: 0 = Disable 1 = Enable	DSK_ ACCESS: 0 = Disable 1 = Enable	LCD_ ACCESS: 0 = Disable 1 = Enable
SYSCFG 4Eh if AEh[7] = 1			**Idle Reload Event Enable Register 1A**				**Default = 00h**
Reserved			GNR7: 0 = Disable 1 = Enable	GNR5: 0 = Disable 1 = Enable	Reserved	Any PCI requests: 0 = Disable 1 = Enable	Reserved
SYSCFG 4Fh			**IDLE_TIMER Register**				**Default = 00h**
Time count byte for IDLE_TIMER: Monitors selected IRQs and EPMIs. Time-out generates PMI#4.							

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG 50h			**PMU Control Register 3**				**Default = 00h**
Software start SMI: 0 = Clear SMI 1 = Start SMI	Reserved	IRQ8 polarity: 0 = Active low 1 = Active high	14.3MHz to 82C700: 0 = Enable 1 = Disable	Write = 1 to start Doze Read = Doze status: 0 = Counting 1 = Timed out	Ready to Resume (RO): 0 = Not in Resume 1 = Ready to Resume	PMU mode (RO): 0 = Nothing pending 1 = Suspend active (clear PMI#6)	Start Suspend (WO): 1 = Enter Suspend mode
SYSCFG 51h			**Beeper Control Register**				**Default = 00h**
Reserved						Beeper control: 00 = No Action 01 = 1kHz 10 = Off 11 = 2kHz	
SYSCFG 52h			**Scratchpad Register 1**				**Default = 00h**
General purpose storage byte: - For CISA Configuration Cycles: Data phase information, low byte							
SYSCFG 53h			**Scratchpad Register 2**				**Default = 00h**
General purpose storage byte. - For CISA Configuration Cycles: Data phase information, high byte							
SYSCFG 54h			**Power Control Latch Register 1**				**Default = 00h**
Enable [3:0] to write latch lines PPWR[3:0]: 0 = Disable 1 = Enable				Read/write data bits for PPWR[3:0]: 0 = Latch output low 1 = Latch output high			
SYSCFG 55h			**Power Control Latch Register 2**				**Default = 0Fh**
Enable [3:0] to write latch lines PPWR[7:4]: 0 = Disable 1 = Enable				Read/write data bits for PPWR[7:4] (Default = 1111): 0 = Latch output low 1 = Latch output high			
SYSCFG 56h			**Reserved**				**Default = 00h**
SYSCFG 57h			**PMU Control Register 4**				**Default = 08h**
Reserved	INTRGRP generates PMI#6: 0 = Disable 1 = Enable	DSK_ACCESS includes FDD: 0 = Yes 1 = No	DSK_ACCESS includes HDD: 0 = Yes 1 = No	LCD video area includes A and B segments: 0 = Disable 1 = Enable	LCD video frame buffer area, use PCIDV0 41h[7:0] and 40h[7:6]: 0 = Disable 1 = Enable	Reserved	

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG 58h			**PMU Event Register 1**				**Default = 00h**
LOWBAT PMI#3 SMI: 00 = Disable 11 = Enable		EPMI1# PMI#2 SMI: 00 = Disable 11 = Enable		EPMI0# PMI#1 SMI: 00 = Disable 11 = Enable		LLOWBAT PMI#0 SMI: 00 = Disable 11 = Enable	
SYSCFG 59h			**PMU Event Register 2**				**Default = 00h**
Allow software SMI: 0 = Disable 1 = Enable	Reload timers on Resume: 0 = No 1 = Yes	Resume INTRGRP PMI#6, Suspend PMI#7 SMI: 00 = Disable 11 = Enable		R_TIMER PMI#5 SMI: 00 = Disable 11 = Enable		IDLE_TIMER PMI#4 SMI: 00 = Disable 11 = Enable	
SYSCFG 5Ah if AEh[7] = 0			**PMU Event Register 3**				**Default = 00h**
GNR1_TIMER PMI#11 GNR1_ACCESS PMI#15: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		KBD_TIMER PMI#10 KBD_ACCESS PMI#14: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		DSK_TIMER PMI#9 DSK_ACCESS PMI#13: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		LCD_TIMER PMI#8 LCD_ACCESS PMI#12: 00 = Disable 01 = Reserved 10 = Reserved 11 = SMI	
SYSCFG 5Ah if AEh[7] = 1			**PMU Event Register 3A**				**Default = 00h**
GNR5_TIMER PMI: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		Reserved					
SYSCFG 5Bh if AEh[7] = 0			**PMU Event Register 4**				**Default = 00h**
Reserved	Global SMI control: 0 = Allow 1 = Mask	Reserved		GNR1 Next Access PMI#15: 0 = Disable 1 = Enable	KBD Next Access PMI#14: 0 = Disable 1 = Enable	DSK Next Access PMI#13: 0 = Disable 1 = Enable	LCD Next Access PMI#12: 0 = Disable 1 = Enable
SYSCFG 5Bh if AEh[7] = 1			**PMU Event Register 4A**				**Default = 00h**
Reserved				GNR5 Next Access PMI#15: 0 = Disable 1 = Enable	Reserved		
SYSCFG 5Ch			**PMI SMI Source Register 1 (Write 1 to Clear)**				**Default = 00h**
PMI#7, Suspend: 0 = Not Active 1 = Active	PMI#6, Resume or INTRGRP: 0 = Not Active 1 = Active	PMI#5, R_TIMER time-out: 0 = Not Active 1 = Active	PMI#4, IDLE_TIMER time-out: 0 = Not Active 1 = Active	PMI#3, LOWBAT: 0 = Not Active 1 = Active	PMI#2, EPMI1#: 0 = Not Active 1 = Active	PMI#1, EPMI0#: 0 = Not Active 1 = Active	PMI#0, LLOWBAT: 0 = Not Active 1 = Active

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG 5Dh if AEh[7] = 0			**PMI SMI Source Register 2 (Write 1 to Clear)**				**Default = 00h**
PMI#15, GNR1_ ACCESS: 0 = None 1 = Active	PMI#14, KBD_ACCESS: 0 = Not Active 1 = Active	PMI#13, DSK_ACCESS: 0 = Not Active 1 = Active	PMI#12, LCD_ACCESS: 0 = Not Active 1 = Active	PMI#11, GNR1_TIMER: 0 = Not Active 1 = Active	PMI#10, KBD_TIMER: 0 = Not Active 1= Active	PMI#9, DSK_TIMER: 0 = Not Active 1 = Active	PMI#8, LCD_TIMER: 0 = Not Active 1 = Active
SYSCFG 5Dh if AEh[7] = 1			**PMI SMI Source Register 2A (Write 1 to Clear)**				**Default = 00h**
PMI#15, GNR5_ ACCESS: 0 = None 1 = Active	Reserved			PMI#11 GNR5_TIMER: 0 = None 1 = Active	Reserved		
SYSCFG 5Eh			**Reserved**				**Default = 00h**
SYSCFG 5Fh			**PMU Control Register 5**				**Default = 00h**
LCD_ACCESS includes ISA bus video access: 0 = Yes 1 = No	LCD_ACCESS includes local (PCI) bus video access: 0 = No 1 = Yes	RSMGRP IRQs can Resume system: 0 = No 1 = Yes	Transitions on RINGI can Resume system: 0 = No 1 = Yes	Number of RINGI transitions to cause Resume			
SYSCFG 60h			**R_Timer Count Register**				**Default = 00h**
Read R_Timer original count							
SYSCFG 61h			**Debounce Register**				**Default = 00h**
LOWBAT, LLOWBAT debounce rate select: 00 = No debounce 01 = 250μs 10 = 8ms 11 = 500ms		SUS/RES# debounce rate select: 00 = Active low, edge-trig'd PMI 01 = Active low, level-controlled PMI 10 = Active high, level-sampled PMI in 16ms 11 = Active high, level-sampled PMI in 32ms (See Section 4.15.8.2, "SUS/ RES and RINGI Events")		PPWR0 auto-toggle in APM STPCLK mode: 0 = No PPWR0 auto-toggle 1 = Auto-toggle PPWR0 on entry & exit from APM STPCLK mode	STPCLK# signal 0 = Disable 1 = Enable	APM STPCLK recovery time: 00 = 120μs 01 = 240μs 10 = 1ms 11 = 2ms	
SYSCFG 62h			**IRQ Doze Register 1**				**Default = 00h**
IRQ13 Doze reset: 0 = Disable 1 = Enable	IRQ8 Doze reset: 0 = Disable 1 = Enable	IRQ7 Doze reset: 0 = Disable 1 = Enable	IRQ12 Doze reset: 0 = Disable 1 = Enable	IRQ5 Doze reset: 0 = Disable 1 = Enable	IRQ4 Doze reset: 0 = Disable 1 = Enable	IRQ3 Doze reset: 0 = Disable 1 = Enable	IRQ0 Doze reset: 0 = Disable 1 = Enable
SYSCFG 63h			**Idle Time-Out Select Register 1 (WO)**				**Default = 00h**
EPMI0# Level-trig'd: 0 = Disable 1 = Enable	IRQ13: 0 = Disable 1 = Enable	IRQ8: 0 = Disable 1 = Enable	IRQ7: 0 = Disable 1 = Enable	IRQ5: 0 = Disable 1 = Enable	IRQ4: 0 = Disable 1 = Enable	IRQ3: 0 = Disable 1 = Enable	IRQ0: 0 = Disable 1 = Enable

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG 64h			**INTRGRP IRQ Select Register 1**				**Default = 00h**
IRQ14: 0 = Disable 1 = Enable	IRQ8: 0 = Disable 1 = Enable	IRQ7: 0 = Disable 1 = Enable	IRQ6: 0 = Disable 1 = Enable	IRQ5: 0 = Disable 1 = Enable	IRQ4: 0 = Disable 1 = Enable	IRQ3: 0 = Disable 1 = Enable	IRQ1: 0 = Disable 1 = Enable
SYSCFG 65h			**Doze Register**				**Default = 00h**
All interrupts to CPU reset Doze mode: 0 = Disable 1 = Enable	Reserved	EPMI0# Doze reset: 0 = Disable 1 = Enable	Recognize SMI during STPCLK#: 0 = No 1 = Yes	IRQ1 Doze reset: 0 = Disable 1 = Enable	EPMI3# Doze reset: 0 = Disable 1 = Enable	EPMI2# Doze reset: 0 = Disable 1 = Enable	EPMI1# Doze reset: 0 = Disable 1 = Enable
SYSCFG 66h			**PMU Control Register 6**				**Default = 00h**
Suspend-to-Normal refresh delay: 0 = None 1 = Three 32KHz CLKs Write to 1 always.	Suspend mode ATCLK frequency: 0 = Derived from PCICLK 1 = 32KHz Setting can be overridden by SYSCFG 79h[0]	Doze type: 0 = Modulate STPCLK# 1 = Keep STPCLK# asserted	Reserved	APM PPWR0 auto-toggle refresh control: 00 = Normal refresh 01 = Refresh pulse is 32KHz 10 = Refresh pulse is 32KHz and engage suspend type DRAM refresh 11 = Reserved Not supported.	Hot docking refresh control. 0 = Normal refresh 1= Refresh pulse is 32KHz and engage Suspend type DRAM refresh Not supported.	STPGNT cycle wait option: 0 = Do not wait 1 = Wait for STPGNT cycle before negating STPCLK#	
SYSCFG 67h			**PMU Control Register 7**				**Default = 00h**
Reserved				Prevent STPCLK# generation by SYSCFG 50h[3] when INTR is active: 0 = Disable 1 = Enable	Normal mode STPCLK# modulation (read returns current STPCLK modulation setting only; STPCLK# modulated by BCLK defined in SYSCFG E6h[7:6]): 000 = No Modulation (STPCLK# = 1) 001 = STPCLK# t_{hi} = 0.75 * 16 BCLKs 010 = STPCLK# t_{hi} = 0.5 * 16 BCLKs 011 = STPCLK# t_{hi} = 0.25 * 16 BCLKs 100 = STPCLK# t_{hi} = 0.125 * 16 BCLKs 101 = STPCLK# t_{hi} = 0.0625 * 16 BCLKs 110 = STPCLK# t_{hi} = 0.03125 * 32 BCLKs 111 = STPCLK# t_{hi} = 0.015625 * 64 BCLKs		
SYSCFG 68h			**Clock Source Register 2**				**Default = 00h**
Clock source for R_TIMER		Clock source for IDLE_TIMER		Resume recovery time: 00 = 8ms 10 = 128ms 01 = 32ms 11 = 30μs **Note:** Ignored if BEh[0] = 1.		PPWR[1:0] auto-toggle on entry and exit from Suspend: 0 = Disable 1 = Enable	
SYSCFG 69h			**R_TIMER Register**				**Default = 00h**
- Time count byte for R_TIMER - starts to count after a non-zero write to this register. - Unlike the other timer registers, a read from this register returns the **current** count. - Time-out generates PMI#5.							

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG 6Ah			**RSMGRP IRQ Register 1**				**Default = 00h**
EPMI1# Resume: 0 = Disable 1 = Enable	EPMI0# Resume: 0 = Disable 1 = Enable	IRQ8 Resume: 0 = Disable 1 = Enable	IRQ7 Resume: 0 = Disable 1 = Enable	IRQ5 Resume: 0 = Disable 1 = Enable	IRQ4 Resume: 0 = Disable 1 = Enable	IRQ3 Resume: 0 = Disable 1 = Enable	IRQ1 Resume: 0 = Disable 1 = Enable
SYSCFG 6Bh			**Resume Source Register**				**Default = 00h**
DRAM Suspend mode refresh type: 0 = Slow refresh (normal) 1 = Self-refresh	PREQ# caused Resume (RO): 0 = No 1 = Yes	CLKRUN# caused Resume (RO): 0 = No 1 = Yes	Reserved: Write as read.	CISA SEL#/ATB# low caused Resume (RO): 0 = No 1 = Yes	SUSP/RSM caused Resume (RO): 0 = No 1 = Yes	RSMGRP caused Resume (RO): 0 = No 1 = Yes	RI caused Resume (RO): 0 = No 1 = Yes
SYSCFG 6Ch			**Scratchpad Register 3**				**Default = 00h**
General purpose storage byte - For CISA Configuration Cycles: Address phase 1 information, low byte							
SYSCFG 6Dh			**Scratchpad Register 4**				**Default = 00h**
General purpose storage byte - For CISA Configuration Cycles: Address phase 1 information, high byte							
SYSCFG 6Eh			**Scratchpad Register 5**				**Default = 00h**
General purpose storage byte - For CISA Configuration Cycles: Address phase 2 information, low byte							
SYSCFG 6Fh			**Scratchpad Register 6**				**Default = 00h**
General purpose storage byte - For CISA Configuration Cycles: Address phase 2 information, high byte							
SYSCFG 70h			**GNR1 Base Address Register 1**				**Default = 00h**
GNR1_ACCESS base address: A[13:6] for memory watchdog or A[15:10] for I/O (right-aligned).							
SYSCFG 71h			**GNR1 Control Register 1**				**Default = FFh**
GNR1_ACCESS mask bits: Mask for A[13:6] for memory watchdog or mask for A[15:10] for I/O (right-aligned).							
SYSCFG 72h			**GNR1 Control Register 2**				**Default = 00h**
GNR1_ACCESS base address: A[5:2] for memory watchdog or ignored for I/O.				GNR1_ACCESS mask bits: Mask for A[5:2] for memory watchdog or mask for A[9:6] for I/O.			
SYSCFG 73h			**GNR2 Base Address Register 1**				**Default = 00h**
GNR2_ACCESS base address: A[13:6] for memory watchdog or A[15:10] for I/O (right-aligned).							
SYSCFG 74h			**GNR2 Control Register 1**				**Default = FFh**
GNR2_ACCESS mask bits: Mask for A[13:6] for memory watchdog or mask for A[15:10] for I/O (right-aligned).							

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG 75h			**GNR2 Control Register 2**				**Default = 00h**
GNR2_ACCESS base address: A[5:2] for memory watchdog or ignored for I/O.				GNR2_ACCESS mask bits: Mask for A[5:2] for memory watchdog or mask for A[9:6] for I/O.			
SYSCFG 76h if AEh[7] = 0			**Doze Reload Select Register 1**				**Default = 0Fh**
LCD_ ACCESS: 0 = DOZE_0 1 = DOZE_1	KBD_ ACCESS: 0 = DOZE_0 1 = DOZE_1	DSK_ ACCESS: 0 = DOZE_0 1 = DOZE_1	HDU_ ACCESS: 0 = DOZE_0 1 = DOZE_1	COM1&2_ ACCESS: 0 = DOZE_0 1 = DOZE_1	LPT_ ACCESS: 0 = DOZE_0 1 = DOZE_1	GNR1_ ACCESS: 0 = DOZE_0 1 = DOZE_1	GNR2_ ACCESS: 0 = DOZE_0 1 = DOZE_1
SYSCFG 76h if AEh[7] = 1			**Doze Reload Select Register 1A**				**Default = 03h**
Reserved		PREQ#: 0 = DOZE_0 1 = DOZE_1	CLKRUN#: 0 = DOZE_0 1 = DOZE_1	Reserved		GNR5: 0 = DOZE_0 1 = DOZE_1	GNR6: 0 = DOZE_0 1 = DOZE_1
SYSCFG 77h			**Doze Reload Select Register 2**				**Default = 00h**
IRQ8: 0 = DOZE_0 1 = DOZE_1	IRQ7: 0 = DOZE_0 1 = DOZE_1	IRQ6: 0 = DOZE_0 1 = DOZE_1	IRQ5: 0 = DOZE_0 1 = DOZE_1	IRQ4: 0 = DOZE_0 1 = DOZE_1	IRQ3: 0 = DOZE_0 1 = DOZE_1	IRQ1: 0 = DOZE_0 1 = DOZE_1	IRQ0: 0 = DOZE_0 1 = DOZE_1
SYSCFG 78h			**Doze Reload Select Register 3**				**Default = 00h**
PCI: 0 = DOZE_0 1 = DOZE_1	IRQ15: 0 = DOZE_0 1 = DOZE_1	IRQ14: 0 = DOZE_0 1 = DOZE_1	IRQ13: 0 = DOZE_0 1 = DOZE_1	IRQ12: 0 = DOZE_0 1 = DOZE_1	IRQ11: 0 = DOZE_0 1 = DOZE_1	IRQ10: 0 = DOZE_0 1 = DOZE_1	IRQ9: 0 = DOZE_0 1 = DOZE_1
SYSCFG 79h			**PMU Control Register 8**				**Default = 00h**
DOZE_1 time-out select: 000 = No delay (Default) 001 = 1ms 010 = 4ms 011 = 16ms 100 = 64ms 101 = 256ms 110 = 1s 111 = 4s			PMI# event triggers exit from Doze mode if the PMI event is enabled to generate SMI:(1) 0 = No 1 = Yes	Reserved	PREQ# wake up Suspend: 0 = Disable 1 = Enable	CLKRUN# wake up Suspend: 0 = Disable 1 = Enable	ATCLK during Suspend: 0 = Run 1 = Stopped (overrides SYSCFG 66h[6])
(1) For example, to let PMI#11 reset the Doze mode without generating SMI to the CPU, SYSCFG 5Ah[7:6] must = 11 and SYSCFG 5Bh[6] must = 1.							
SYSCFG 7Ah			**GNR3 Base Address Register 1**				**Default = 00h**
GNR3_ACCESS base address: A[13:6] for memory watchdog or A[15:10] for I/O (right-aligned).							
SYSCFG 7Bh			**GNR3 Control Register 1**				**Default = FFh**
GNR3_ACCESS mask bits: Mask for A[13:6] for memory watchdog or mask for A[15:10] for I/O (right-aligned).							
SYSCFG 7Ch			**GNR3 Control Register 2**				**Default = 00h**
GNR3_ACCESS base address: A[5:2] for memory watchdog or ignored for I/O.				GNR3_ACCESS mask bits: Mask for A[5:2] for memory watchdog or mask for A[9:6] for I/O.			
SYSCFG 7Dh			**GNR4 Base Address Register 1**				**Default = 00h**
GNR4_ACCESS base address: A[13:6] for memory watchdog or A[15:10] for I/O (right-aligned).							

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG 7Eh			**GNR4 Control Register 1**				**Default = FFh**
GNR4_ACCESS mask bits: Mask for A[13:6] for memory watchdog or mask for A[15:10] for I/O (right-aligned).							
SYSCFG 7Fh			**GNR4 Control Register 2**				**Default = 00h**
GNR4_ACCESS base address: A[5:2] for memory watchdog or ignored for I/O.				GNR4_ACCESS mask bits: Mask for A[5:2] for memory watchdog or mask for A[9:6] for I/O.			
SYSCFG 80h			**ICW1 Shadow Register for INTC1**				**Default = 00h**
SYSCFG 81h			**ICW2 Shadow Register for INTC1**				**Default = 00h**
SYSCFG 82h			**ICW3 Shadow Register for INTC1**				**Default = 00h**
SYSCFG 83h			**ICW4 Shadow Register for INTC1**				**Default = 00h**
SYSCFG 84h			**DMA In-Progress Register (RO)**				**Default = 00h**
Ch. 7 DMA in progress: 0 = No 1 = Possibly	Ch. 6 DMA in progress: 0 = No 1 = Possibly	Ch. 5 DMA in progress: 0 = No 1 = Possibly	DMAC2 byte pointer flip-flop. 0 = Cleared 1 = Set	Ch. 3 DMA in progress: 0 = No 1 = Possibly	Ch. 2 DMA in progress: 0 = No 1 = Possibly	Ch. 1 DMA in progress: 0 = No 1 = Possibly	Ch. 0 DMA in progress: 0 = No 1 = Possibly
SYSCFG 85h			**OCW2 Shadow Register for INTC1**				**Default = 00h**
SYSCFG 86h			**OCW3 Shadow Register for INTC1**				**Default = 00h**
SYSCFG 87h			**Reserved**				**Default = 00h**
SYSCFG 88h			**ICW1 Shadow Register for INTC2**				**Default = 00h**
SYSCFG 89h			**ICW2 Shadow Register for INTC2**				**Default = 00h**
SYSCFG 8Ah			**ICW3 Shadow Register for INTC2**				**Default = 00h**
SYSCFG 8Bh			**ICW4 Shadow Register for INCT2**				**Default = 00h**
SYSCFG 8Ch			**Reserved**				**Default = 00h**
SYSCFG 8Dh			**OCW2 Shadow Register for INTC2**				**Default = 00h**
SYSCFG 8Eh			**OCW3 Shadow Register for INTC2**				**Default = 00h**
SYSCFG 8Fh			**Reserved**				**Default = 00h**
SYSCFG 90h			**Timer Channel 0 Low Byte Register: A[7:0]**				**Default = 00h**
SYSCFG 91h			**Timer Channel 0 High Byte Register: A[15:8]**				**Default = 00h**
SYSCFG 92h			**Timer Channel 1 Low Byte Register: A[7:0]**				**Default = 00h**
SYSCFG 93h			**Timer Channel 1 High Byte Register: A[15:8]**				**Default = 00h**
SYSCFG 94h			**Timer Channel 2 Low Byte Register: A[7:0]**				**Default = 00h**
SYSCFG 95h			**Timer Channel 2 High Byte Register: A[15:8]**				**Default = 00h**

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG 96h			**Write Counter High/Low Byte Latch (RO)**				**Default = xxh**
Unused	Unused	Timer Ch. 2 read LSB toggle bit	Timer Ch. 1 read LSB toggle bit	Timer Ch. 0 read LSB toggle bit	Timer Ch. 2 write LSB toggle bit	Timer Ch. 1 write LSB toggle bit	Timer Ch. 0 write LSB toggle bit
SYSCFG 97h			**Reserved**				**Default = 00h**
SYSCFG 98h			**RTC Index Shadow Register (RO)**				**Default = xxh**
NMI enable setting	CMOS RAM Index last written						
SYSCFG 99h			**Interrupt Request Register for INCT1 (RO)**				**Default = xxh**
IRQ7 pending: 0 = No 1 = Yes	IRQ6 pending: 0 = No 1 = Yes	IRQ5 pending: 0 = No 1 = Yes	IRQ4 pending: 0 = No 1 = Yes	IRQ3 pending: 0 = No 1 = Yes	IRQ2 pending: 0 = No 1 = Yes	IRQ1 pending: 0 = No 1 = Yes	IRQ0 pending: 0 = No 1 = Yes
SYSCFG 9Ah			**Interrupt Request Register for INCT2 (RO)**				**Default = xxh**
IRQ15 pending: 0 = No 1 = Yes	IRQ14 pending: 0 = No 1 = Yes	IRQ13 pending: 0 = No 1 = Yes	IRQ12 pending: 0 = No 1 = Yes	IRQ11 pending: 0 = No 1 = Yes	IRQ10 pending: 0 = No 1 = Yes	IRQ9 pending: 0 = No 1 = Yes	IRQ8 pending: 0 = No 1 = Yes
SYSCFG 9Bh			**3F2h + 3F7h Shadow Register**				**Default = 00h**
Shadows 3F2h[7] "Mode Select" bit	Shadows 3F7h[1] "Disk Type" bit 1	Shadows 3F2h[5] "Drive 2 Motor" bit	Shadows 3F2h[4] "Drive 1 Motor" bit	Shadows 3F2h[3] "DMA Enable" bit	Shadows 3F2h[2] "Soft Reset" bit	Shadows 3F7h[0] "Disk Type" bit 0	Shadows 3F2h[0] "Drive Select" bit
SYSCFG 9Ch			**372h + 377h Shadow Register**				**Default = 00h**
Shadows 372h[7] "Mode Select" bit	Shadows 377h[1] "Disk Type" bit 1	Shadows 372h[5] "Drive 2 Motor" bit	Shadows 372h[4] "Drive 1 Motor" bit	Shadows 372h[3] "DMA Enable" bit	Shadows 372h[2] "Soft Reset" bit	Shadows 377h[0] "Disk Type" bit 0	Shadows 372h[0] "Drive Select" bit
SYSCFG 9Dh-9Eh			**Reserved**				**Default = 00h**
SYSCFG 9Fh			**Port 064h Shadow Register**				**Default = 00h**
Shadows I/O writes to Port 064h bits [7:0] (regardless of whether KBDCS# is inhibited). - In this way, when an SMI occurs between a Port 064h write and the subsequent write to Port 060h, SMM code can access the keyboard controller as needed and then simply restore the Port 064h value just before leaving SMM.							
SYSCFG A0h			**Feature Control Register 1**				**Default = 80h**
16-bit I/O decoding: 0 = Disable 1 = Enable	Reserved						
SYSCFG A1h			**Feature Control Register 2**				**Default = 00h**
Reserved					Emerg. over-temp sense: 0 = Disable 1 = Enable	Reserved	EPMI[1:0]# status latch: 0 = Dynamic 1 = Latched

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG A2h if AEh[7] = 0			**IRQ Doze Register 2**				**Default = 00h**
PCI bus I/O access Doze reset: 0 = Disable 1 = Enable	PCI memory access Doze reset: 0 = Disable 1 = Enable	IRQ15 Doze reset: 0 = Disable 1 = Enable	IRQ14 Doze reset: 0 = Disable 1 = Enable	IRQ11 Doze reset: 0 = Disable 1 = Enable	IRQ10 Doze reset: 0 = Disable 1 = Enable	IRQ9 Doze reset: 0 = Disable 1 = Enable	IRQ6 Doze reset: 0 = Disable 1 = Enable
SYSCFG A2h if AEh[7] = 1			**IRQ Doze Register 2A**				**Default = 00h**
PREQ# Doze reset: 0 = Disable 1 = Enable	CLKRUN# Doze reset: 0 = Disable 1 = Enable	Reserved					
SYSCFG A3h			**Idle Time-Out Select Register 2 (WO)**				**Default = 00h**
IRQ15: 0 = Disable 1 = Enable	IRQ14: 0 = Disable 1 = Enable	IRQ12: 0 = Disable 1 = Enable	IRQ11: 0 = Disable 1 = Enable	IRQ10: 0 = Disable 1 = Enable	IRQ9: 0 = Disable 1 = Enable	IRQ6: 0 = Disable 1 = Enable	IRQ1: 0 = Disable 1 = Enable
SYSCFG A4h			**INTRGRP IRQ Select Register 2**				**Default = 00h**
Test Bit: Write as 0	IRQ15: 0 = Disable 1 = Enable	IRQ13: 0 = Disable 1 = Enable	IRQ12: 0 = Disable 1 = Enable	IRQ11: 0 = Disable 1 = Enable	IRQ10: 0 = Disable 1 = Enable	IRQ9: 0 = Disable 1 = Enable	IRQ0: 0 = Disable 1 = Enable
SYSCFG A5h			**Thermal Management Register 1**				**Default = 00h**
Thermal Mgmt.: 0 = Disable 1 = Enable	TEMPDET Variation - As temperature increases, frequency: 0 = Decreases 1 = Increases	Level 2 STPCLK# modulation rate - Sets the stop clock throttling rate when temperature enters second (overtemp) range: 000 = No modulation (STPCLK# = 1) 001 = STPCLK# t_{hi} = 0.75 * 16 BCLKs 010 = STPCLK# t_{hi} = 0.5 * 16 BCLKs 011 = STPCLK# t_{hi} = 0.25 * 16 BCLKs 100 = STPCLK# t_{hi} = 0.125 * 16 BCLKs 101 = STPCLK# t_{hi} = 0.0625 * 16 BCLKs 110 = STPCLK# t_{hi} = 0.03125 * 32 BCLKs 111 = STPCLK# t_{hi} = 0.015625 * 64 BCLKs			Level 1 STPCLK# modulation rate - Sets the stop clock throttling rate when temperature enters first (high temp) range: 000 = No modulation (STPCLK# = 1) 001 = STPCLK# t_{hi} = 0.75 * 16 BCLKs 010 = STPCLK# t_{hi} = 0.5 * 16 BCLKs 011 = STPCLK# t_{hi} = 0.25 * 16 BCLKs 100 = STPCLK# t_{hi} = 0.125 * 16 BCLKs 101 = STPCLK# t_{hi} = 0.0625 * 16 BCLKs 110 = STPCLK# t_{hi} = 0.03125 * 32 BCLKs 111 = STPCLK# t_{hi} = 0.015625 * 64 BCLKs		
Note: Once thermal management has been enabled (bit 7 = 1), none of the thermal management registers can be overwritten.							
SYSCFG A6h			**Thermal Management Register 2**				**Default = 00h**
LOFREQ[7:0]: Low frequency limit low byte							
SYSCFG A7h			**Thermal Management Register 3**				**Default = 00h**
LOFREQ[15:8]: Low frequency limit high byte							
SYSCFG A8h			**Thermal Management Register 4**				**Default = 00h**
HIFREQ[7:0]: High frequency limit low byte							
SYSCFG A9h			**Thermal Management Register 5**				**Default = 00h**
HIFREQ[15:8]: High frequency limit high byte							

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG AAh			**Thermal Management Register 6**				**Default = 00h**
Emergency Overtemp Sensor STPCLK# Modulation Rate: 000 = No modulation (STPCLK# = 1) 001 = STPCLK# t_{hi} = 0.75 * 16 BCLKs 010 = STPCLK# t_{hi} = 0.5 * 16 BCLKs 011 = STPCLK# t_{hi} = 0.25 * 16 BCLKs 100 = STPCLK# t_{hi} = 0.125 * 16 BCLKs 101 = STPCLK# t_{hi} = 0.0625 * 16 BCLKs 110 = STPCLK# t_{hi} = 0.03125 * 32 BCLKs 111 = STPCLK# t_{hi} = 0.015625 * 64 BCLKs			THMIN pin polarity: 0 = High 1 = Low	EPMI trigger for thermal mgmt. 00 = EPMI0# 01 = EPMI1# 10 = EPMI2# 11 = EPMI3# Also see bit 1.		THMIN input: 0 = THMIN 1 = EPMI indicated in bits [3:2]	HDI input: 0 = HDI 1 = EPMI indicated by SYSCFG F0h[1:0]
SYSCFG ABh			**Power Control Latch Register 3**				**Default = 00h**
Enable [3:0] to write latch lines PPWR[11:8]: 0 = Disable 1 = Enable				Read/write data bits for PPWR[11:8]: 0 = Latch output low 1 = Latch output high			
SYSCFG ACh			**Reserved**				**Default = 00h**
SYSCFG ADh			**Feature Control Register 3**				**Default = 00h**
Reserved		CPU power state in Suspend: 0 = Powered 1 = 0 Volt	PCIIDE responds as: 0 = Device 14h, Function 0 1 = Device 01h, Function 1	INIT operation: 0 = Normal 1 = Toggle on Resume	PCIIDE Device ID: 0 = D568h 1 = D721h	Reserved	
SYSCFG AEh			**GNR_ACCESS Feature Register 1**				**Default = 03h**
GNR set select: 0 = GNR1-4 1 = GNR5-8	Reserved	GNR2 cycle decode type: 0 = I/O 1 = Memory	GNR1 cycle decode type: 0 = I/O 1 = Memory	GNR2 base address: A0 (I/O) A14 (Memory)	GNR1 base address: A0 (I/O) A14 (Memory)	GNR2 mask bit: A0 (I/O) A14 (Memory)	GNR1 mask bit: A0 (I/O) A14 (Memory)
SYSCFG AFh-B0h			**Reserved**				**Default = 00h**
SYSCFG B1h			**RSMGRP IRQ Register 2**				**Default = 00h**
EPMI3# Resume: 0 = Disable 1 = Enable	EPMI2# Resume: 0 = Disable 1 = Enable	IRQ15 Resume: 0 = Disable 1 = Enable	IRQ14 Resume: 0 = Disable 1 = Enable	IRQ12 Resume: 0 = Disable 1 = Enable	IRQ11 Resume: 0 = Disable 1 = Enable	IRQ10 Resume: 0 = Disable 1 = Enable	IRQ9 Resume: 0 = Disable 1 = Enable
SYSCFG B2h if AEh[7] = 0			**Clock Source Register 3**				**Default = 00h**
Clock source for HDU_TIMER		Clock source for COM2_TIMER		Clock source for COM1_TIMER		Clock source for GNR2_TIMER	
SYSCFG B2h if AEh[7] = 1			**Clock Source Register 3A**				**Default = 00h**
Reserved						Clock source for GNR6_TIMER	

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG B3h			**Chip Select Cycle Type Register**				**Default = 00h**
GPCS3# ROM width: 0 = 8-bit 1 = 16-bit	GPCS2# ROM width: 0 = 8-bit 1 = 16-bit	GPCS1# ROM width: 0 = 8-bit 1 = 16-bit	GPCS0# ROM width: 0 = 8-bit 1 = 16-bit	GPCS3# cycle type: 0 = I/O 1 = ROMCS	GPCS2# cycle type: 0 = I/O 1 = ROMCS	GPCS1# cycle type: 0 = I/O 1 = ROMCS	GPCS0# cycle type: 0 = I/O 1 = ROMCS
SYSCFG B4h			**HDU_TIMER Register**				**Default = 00h**
Time count byte for HDU_TIMER: Monitors HDU_ACCESS. Time-out generates PMI#19.							
SYSCFG B5h			**COM1_TIMER Register**				**Default = 00h**
Time count byte for COM1_TIMER: Monitors COM1_ACCESS. Time-out generates PMI#17.							
SYSCFG B6h			**COM2_TIMER Register**				**Default = 00h**
Time count byte for COM2_TIMER: Monitors COM2_ACCESS. Time-out generates PMI#18.							
SYSCFG B7h if AEh[7] = 0			**GNR2_TIMER Register**				**Default = 00h**
Time count byte for GNR2_TIMER: Monitors GNR2_ACCESS. Time-out generates PMI#16.							
SYSCFG B7h if AEh[7] = 1			**GNR6_TIMER Register**				**Default = 00h**
Time count byte for GNR6_TIMER: Monitors GNR6_ACCESS. Time-out generates PMI#16.							
SYSCFG B8h if AEh[7] = 0			**GNR2 Base Address Register**				**Default = 00h**
GNR2_ACCESS base address: A[8:1] (I/O) or A[22:15] (Memory)							
SYSCFG B8h if AEh[7] = 1			**GNR6 Base Address Register**				**Default = 00h**
GNR6_Timer base address: A[8:1] (I/O)							
SYSCFG B9h if AEh[7] = 0			**GNR2 Control Register**				**Default = 00h**
GNR2 base address: A9 (I/O) A23 (Memory)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	GNR2 mask bits for address A[5:1] (I/O) or A[19:15] memory: A 1 in a particular bit means that the corresponding bit at B8h[4:0] is not compared. This is used to determine address block size.				
SYSCFG B9h if AEh[7] = 1			**GNR6 Control Register**				**Default = 00h**
GNR6 base address: A9 (I/O)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	GNR6 mask bits for address A[5:1] (I/O)				
SYSCFG BAh			**Chip Select 2 Base Address Register**				**Default = 00h**
GPCS2# base address: A[8:1] (I/O) or A[22:15] (Memory)							
SYSCFG BBh			**Chip Select 2 Control Register**				**Default = 00h**
GPCS2# base address: A9 (I/O) A23 (Memory)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	Chip select active: 0 = w/Cmd 1 = before ALE	GPCS2# mask bits for address A[4:1] (I/O) or A[18:15] memory: A 1 in a particular bit means that the corresponding bit at SYSCFG BAh[3:0] is not compared. This is used to determine address block size.			
SYSCFG BCh			**Chip Select 3 Base Address Register**				**Default = 00h**
GPCS3# base address: A[8:1] (I/O) or A[22:15] (Memory)							

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG BDh			**Chip Select 3 Control Register**				**Default = 00h**
GPCS3# base address: A9 (I/O) A23 (Memory)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	Chip select active: 0 = w/Cmd 1 = before ALE	GPCS3# mask bits for address A[4:1] (I/O) or A[18:15] memory: A 1 in a particular bit means that the corresponding bit at SYSCFG BCh[3:0] is not compared. This is used to determine address block size.			
SYSCFG BEh if AEh[7] = 0			**Idle Reload Event Enable Register 2**				**Default = 00h**
GPCS3#_ ACCESS: 0 = Disable 1 = Enable	GPCS2#_ ACCESS: 0 = Disable 1 = Enable	COM2_ ACCESS: 0 = Disable 1 = Enable	COM1_ ACCESS: 0 = Disable 1 = Enable	GNR2_ ACCESS: 0 = Disable 1 = Enable	HDU_ ACCESS: 0 = Disable 1 = Enable	GNR4_ ACCESS: 0 = Disable 1 = Enable	Override SYSCFG 68h[3:2]: 0 = No 1 = Recover time 1s
SYSCFG BEh if AEh[7] = 1			**Idle Reload Event Enable Register 2A**				**Default = 00h**
Reserved				GNR6_ ACCESS: 0 = Disable 1 = Enable	Reserved	GNR8_ ACCESS: 0 = Disable 1 = Enable	Reserved
SYSCFG BFh			**Chip Select Granularity Register**				**Default = 0Fh**
GPCS3# base address: A0 (I/O) A14 (Memory)	GPCS2# base address: A0 (I/O) A14 (Memory)	GPCS1# base address: A0 (I/O) A14 (Memory)	GPCS0# base address: A0 (I/O) A14 (Mem.)	GPCS3# mask bit: A0 (I/O) A14 (Memory)	GPCS2# mask bit: A0 (I/O) A14 (Memory)	GPCS1# mask bit: A0 (I/O) A14 (Memory)	GPCS0# mask bit: A0 (I/O) A14 (Memory)
SYSCFG C0h-D4h			**Reserved**				**Default = 00h**
SYSCFG D5h			**X Bus Positive Decode Register**				**Default = 00h**
IPC Registers(1): 00 = Reserved 01 = Positive decode 10 = Reserved 11 = Reserved		RTCRD/WR#, RTCAS I/O Ports 70h-71h: 00 = Reserved 01 = Positive decode 10 = Reserved 11 = Reserved		KBDCS# I/O Ports 60, 64, 62, 66, 92h: 00 = Reserved 01 = Positive decode 10 = Reserved 11 = Reserved		ROMCS# memory segments C000h-F000h: 00 = Reserved 01 = Positive decode 10 = Reserved 11 = Reserved	
(1) I/O 20, 21, A0, A1, 40-43, 00-0F, C0-CF, page registers, high page registers, 22, 24, 23 if Index = 01h)							
SYSCFG D6h			**PMU Control Register 9**				**Default = 00h**
DSK_ACCESS: 0 = 3F5h only 1 = All FDC Ports (3F2,4,5,7 & 372,4,5,7h)	DMA trap PMI#28 SMI: 0 = Disable 1 = Enable	DMAC1 byte pointer flip-flop (RO): 0 = Cleared 1 = Set	APM doze exit PMI#35: 0 = Disable 1 = Enable	SBHE# status trap (RO)	I/O port access trapped (RO): 0 = I/O read 1 = I/O write	Access trap bit A9 (RO)	Access trap bit A8 (RO)
SYSCFG D7h			**Access Port Address Register 1**				**Default = 00h**
Access trap address bits A[7:0]: - These bits, along with SYSCFG D6h[1:0] and SYSCFG EBh[7:0] provide the 16-bit address of the port access that caused the SMI trap. - SYSCFG D6h[2] indicates whether an I/O read or an I/O write access was trapped. - SYSCFG D6h[3] gives the status of the SBHE# signal for the I/O instruction that was trapped.							

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG D8h if AEh[7] = 0			**PMU Event Register 5**				**Default = 00h**
HDU_TIMER PMI#19 HDU_ACCESS PMI#23: 00 = Disable 01 = Reserved 01 = Reserved 11 = SMI		COM2_TIMER PMI#18 COM2_ACCESS PMI#22: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		COM1_TIMER PMI#17 COM1_ACCESS PMI#21: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		GNR2_TIMER PMI#16 GNR2_ACCESS PMI#20: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI	
SYSCFG D8h if AEh[7] = 1			**PMU Event Register 5A**				**Default = 00h**
Reserved						GNR6_TIMER PMI#16 GNR6_ACCESS PMI#20: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI	
SYSCFG D9h			**PMU Event Register 6**				**Default = 00h**
DOZE_TIMER PMI#27 SMI: 00 = Disable 01 = Enable DOZE_0 10 = Enable DOZE_1 11 = Enable both		RINGI PMI#26 SMI: 00 = Disable 11 = Enable		EPMI3# cool-down clocking PMI#25 SMI: 00 = Disable 11 = Enable		EPMI2# PMI#24 SMI: 00 = Disable 11 = Enable	
SYSCFG DAh			**Power Management Event Status Register (RO)**				**Default = 00h**
Reserved		LOWBAT state: 0 = Low 1 = High	LLOWBAT state: 0 = Low 1 = High	EPMI3# state: 0 = Low 1 = High	EPMI2# state: 0 = Low 1 = High	EPMI1# state: 0 = Low 1 = High	EPMI0# state: 0 = Low 1 = High
SYSCFG DBh if AEh[7] = 0			**Next Access Event Generation Register 1**				**Default = 00h**
I/O blocking control: 0 = Block I/O on Next Access trap 1 = Unblock	SMI on cool-down clocking entry/exit: 0 = Disable 1 = Enable	EPMI3# pin polarity: 0 = Active high 1 = Active low	EPMI2# pin polarity: 0 = Active high 1 = Active low	HDU_ ACCESS PMI#23 on Next Access: 0 = No 1 = Yes	COM2_ ACCESS PMI#22 on Next Access: 0 = No 1 = Yes	COM1_ ACCESS PMI#21 on Next Access: 0 = No 1 = Yes	GNR2_ ACCESS PMI#20 on Next Access: 0 = No 1 = Yes
SYSCFG DBh if AEh[7] = 1			**Next Access Event Generation Register 1A**				**Default = 00h**
Reserved							GNR6_ ACCESS PMI#20 on Next Access: 0 = No 1 = Yes

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG DCh if AEh[7] = 0			**PMU SMI Source Register 1 (Write 1 to Clear)**				**Default = 00h**
PMI#23, HDU_ ACCESS: 0 = Inactive 1 = Active	PMI#22, COM2_ ACCESS: 0 = Inactive 1 = Active	PMI#21, COM1_ ACCESS: 0 = Inactive 1 = Active	PMI#20, GNR2_ ACCESS: 0 = Inactive 1 = Active	PMI#19, HDU_ TIMER: 0 = Inactive 1 = Active	PMI#18, COM2_ TIMER: 0 = Inactive 1 = Active	PMI#17, COM1_ TIMER: 0 = Inactive 1 = Active	PMI#16, GNR2_ TIMER: 0 = Inactive 1 = Active
SYSCFG DCh if AEh[7] = 1			**PMU SMI Source Register 1A (Write 1 to Clear)**				**Default = 00h**
Reserved			PMI#20, GNR6_ ACCESS: 0 = Clear 1 = Active	Reserved			PMI#16, GNR6_ TIMER: 0 = Clear 1 = Active
SYSCFG DDh			**PMU SMI Source Register 2 (Write 1 to Clear)**				**Default = 00h**
PMI#39, PCI retry limit: 0 = Inactive 1 = Active	PMI#38, CISA/PCI IRQ driveback trap: 0 = Inactive 1 = Active	PMI#37, DMA_ ACCESS: 0 = Inactive 1 = Active	PMI#28, DMA Request: 0 = Inactive 1 = Active	PMI#27, DOZE_ TIMER: 0 = Inactive 1 = Active	PMI#26, RINGI: 0 = Inactive 1 = Active	PMI#25, EPMI3# pin/ cool-down clocking: 0 = Inactive 1 = Active	PMI#24, EPMI2# pin: 0 = Inactive 1 = Active
SYSCFG DDh - FS ACPI Version			**PMU SMI Source Register 2 (Write 1 to Clear)**				**Default = 00h**
PMI#39, ACPI SMI: 0 = Inactive 1 = Active	PMI#38, CISA/PCI IRQ driveback trap: 0 = Inactive 1 = Active	PMI#37, DMA_ ACCESS: 0 = Inactive 1 = Active	PMI#28, DMA Request: 0 = Inactive 1 = Active	PMI#27, DOZE_ TIMER: 0 = Inactive 1 = Active	PMI#26, RINGI: 0 = Inactive 1 = Active	PMI#25, EPMI3# pin/ cool-down clocking: 0 = Inactive 1 = Active	PMI#24, EPMI2# pin: 0 = Inactive 1 = Active
SYSCFG DEh if AEh[7] = 0			**Current Access Event Generation Register 1**				**Default = 00h**
HDU_ ACCESS PMI#23 on Current Access: 0 = No 1 = Yes	COM2_ ACCESS PMI#22 on Current Access: 0 = No 1 = Yes	COM1_ ACCESS PMI#21 on Current Access: 0 = No 1 = Yes	GNR2_ ACCESS PMI#20 on Current Access: 0 = No 1 = Yes	GNR1_ ACCESS PMI#15 on Current Access: 0 = No 1 = Yes	KBD_ ACCESS PMI#14 on Current Access: 0 = No 1 = Yes	DSK_ ACCESS PMI#13 on Current Access: 0 = No 1 = Yes	LCD_ ACCESS PMI#12 on Current Access: 0 = No 1 = Yes
SYSCFG DEh if AEh[7] = 1			**Current Access Event Generation Register 1A**				**Default = 00h**
Reserved			GNR6_ ACCESS PMI#20 on Current Access: 0 = No 1 = Yes	Reserved			

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG DFh if AEh[7] = 0			**Activity Tracking Register 1**				**Default = 00h**
HDU_ ACCESS activity: 0 = No 1 = Yes	COM2_ ACCESS activity: 0 = No 1 = Yes	COM1_ ACCESS activity: 0 = No 1 = Yes	GNR2_ ACCESS activity: 0 = No 1 = Yes	GNR1_ ACCESS activity: 0 = No 1 = Yes	KBD_ ACCESS activity: 0 = No 1 = Yes	DSK_ ACCESS activity: 0 = No 1 = Yes	LCD_ ACCESS activity: 0 = No 1 = Yes
SYSCFG DFh if AEh[7] = 1			**Activity Tracking Register 1A**				**Default = 00h**
Reserved			GNR6_ ACCESS activity: 0 = No 1 = Yes	GNR5_ ACCESS activity: 0 = No 1 = Yes	Reserved		
SYSCFG E0h if AEh[7] = 0			**Activity Tracking Register 2**				**Default = 00h**
Reserved						GNR4_ ACCESS activity: 0 = No 1 = Yes	GNR3_ ACCESS activity: 0 = No 1 = Yes
SYSCFG E0h if AEh[7] = 1			**Activity Tracking Register 2A**				**Default = 00h**
Reserved						GNR8_ ACCESS activity: 0 = No 1 = Yes	GNR7_ ACCESS activity: 0 = No 1 = Yes
SYSCFG E1h if AEh[7] = 0			**GNR3 Base Address Register**				**Default = 00h**
GNR3_ACCESS base address: A[8:1] (I/O) or A[22:15] (Memory)							
SYSCFG E1h if AEh[7] = 1			**GNR7 Base Address Register**				**Default = 00h**
GNR7_ACCESS base address: A[8:1] (I/O)							
SYSCFG E2h if AEh[7] = 0			**GNR3 Control Register**				**Default = 00h**
GNR3 base address: A9 (I/O) A23 (Memory)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	GNR3 mask bits for address A[5:1] (I/O) or A[19:15] memory: A 1 in a particular bit means that the corresponding bit at SYSCFG E1h[4:0] is not compared. This is used to determine address block size.				
SYSCFG E2h if AEh[7] = 1			**GNR7 Control Register**				**Default = 00h**
GNR7 base address: A9 (I/O)	Write decode: 0 = Disable 1 = Enable	Read decode: 0 = Disable 1 = Enable	GNR7 mask bits for address A[5:1] (I/O)				
SYSCFG E3h if AEh[7] = 0			**GNR4 Base Address Register**				**Default = 00h**
GNR4_ACCESS base address: A[8:1] (I/O) or A[22:15] (Memory)							
SYSCFG E3h if AEh[7] = 1			**GNR8 Base Address Register**				**Default = 00h**
GNR8_ACCESS base address: A[8:1] (I/O)							

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="3">SYSCFG E4h <u>if AEh[7] = 0</u></td><td colspan="3">GNR4 Control Register</td><td colspan="2">Default = 00h</td></tr>
<tr><td>GNR4 base address:
A9 (I/O)
A23 (Memory)</td><td>Write decode:
0 = Disable
1 = Enable</td><td>Read decode:
0 = Disable
1 = Enable</td><td colspan="5">GNR4 mask bits for address A[5:1] (I/O) or A[19:15] memory:
A 1 in a particular bit means that the corresponding bit at SYSCFG E3h[4:0] is not compared. This is used to determine address block size.</td></tr>
<tr><td colspan="3"><u>SYSCFG E4h if AEh[7] = 1</u></td><td colspan="3"><u>GNR8 Control Register</u></td><td colspan="2"><u>Default = 00h</u></td></tr>
<tr><td><u>GNR8 base address:</u>
<u>A9 (I/O)</u></td><td><u>Write decode:</u>
<u>0 = Disable</u>
<u>1 = Enable</u></td><td><u>Read decode:</u>
<u>0 = Disable</u>
<u>1 = Enable</u></td><td colspan="5"><u>GNR8 mask bits for address A[5:1] (I/O)</u></td></tr>
<tr><td colspan="3">SYSCFG E5h</td><td colspan="3">GNR_ACCESS Feature Register 2</td><td colspan="2">Default = 03h</td></tr>
<tr><td>Reserved</td><td><u>Reserved</u></td><td>GNR4 cycle decode type:
0 = I/O
1 = Memory</td><td>GNR3 cycle decode Type:
0 = I/O
1 = Memory</td><td>GNR4 base address:
A0 (I/O)
A14 (Memory)</td><td>GNR3 base address:
A0 (I/O)
A14 (Memory)</td><td>GNR4 mask bit:
A0 (I/O)
A14 (Memory)</td><td>GNR3 mask bit:
A0 (I/O)
A14 (Memory)</td></tr>
<tr><td colspan="3">SYSCFG E6h <u>if AEh[7] = 0</u></td><td colspan="3">Clock Source Register 4</td><td colspan="2">Default = 70h</td></tr>
<tr><td colspan="2">BCLK source for STPCLK# modulation (For normal mode, Doze mode, thermal mgmt):
00 = 4KHz
01 = 32KHz (Default)
10 = 450KHz
11 = 900KHz</td><td>GNR4_ ACCESS:
0 = DOZE_0
1 = DOZE_1</td><td>GNR3_ ACCESS:
0 = DOZE_0
1 = DOZE_1</td><td colspan="2">Clock source for GNR4_TIMER</td><td colspan="2">Clock source for GNR3_TIMER</td></tr>
<tr><td colspan="3"><u>SYSCFG E6h if AEh[7] = 1</u></td><td colspan="3"><u>Clock Source Register 4A</u></td><td colspan="2"><u>Default = 70h</u></td></tr>
<tr><td colspan="2"><u>Reserved</u></td><td><u>GNR8_ ACCESS:</u>
<u>0 = DOZE_0</u>
<u>1 = DOZE_1</u></td><td><u>GNR7_ ACCESS:</u>
<u>0 = DOZE_0</u>
<u>1 = DOZE_1</u></td><td colspan="2"><u>Clock source for GNR8_TIMER</u></td><td colspan="2"><u>Clock source for GNR7_TIMER</u></td></tr>
<tr><td colspan="3">SYSCFG E7h <u>if AEh[7] = 0</u></td><td colspan="3">GNR3_TIMER Register</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="8">Time count byte for GNR3_TIMER: Monitors GNR3_ACCESS. Time-out generates PMI#29.</td></tr>
<tr><td colspan="3"><u>SYSCFG E7h if AEh[7] = 1</u></td><td colspan="3"><u>GNR7_TIMER Register</u></td><td colspan="2"><u>Default = 00h</u></td></tr>
<tr><td colspan="8"><u>Time count byte for GNR7_TIMER: Monitors GNR7_ACCESS. Time-out generates PMI#29.</u></td></tr>
<tr><td colspan="3">SYSCFG E8h <u>if AEh[7] = 0</u></td><td colspan="3">GNR4_TIMER Register</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="8">Time count byte for GNR4_TIMER: Monitors GNR4_ACCESS. Time-out generates PMI#30.</td></tr>
<tr><td colspan="3"><u>SYSCFG E8h if AEh[7] = 1</u></td><td colspan="3"><u>GNR8_TIMER Register</u></td><td colspan="2"><u>Default = 00h</u></td></tr>
<tr><td colspan="8"><u>Time count byte for GNR8_TIMER: Monitors GNR8_ACCESS. Time-out generates PMI#30.</u></td></tr>
</table>

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG E9h if AEh[7] = 0			**PMU Event Register 7**				**Default = 00h**
GNR4_TIMER PMI#30 GNR4_ACCESS PMI#32: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		GNR3_TIMER PMI#29 GNR3_ACCESS PMI#31: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		GNR4_ ACCESS PMI#32 on Current Access: 0 = No 1 = Yes	GNR4_ ACCESS PMI#32 on Next Access: 0 = No 1 = Yes	GNR3_ ACCESS PMI#31 on Current Access: 0 = No 1 = Yes	GNR3_ ACCESS PMI#31 on Next Access: 0 = No 1 = Yes
SYSCFG E9h if AEh[7] = 1			**PMU Event Register 7A**				**Default = 00h**
GNR8_TIMER PMI#30 GNR8_ACCESS PMI#32: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		GNR7_TIMER PMI#29 GNR7_ACCESS PMI#31: 00 = Disable 01 = Positive decode 10 = Positive decode, SMI 11 = SMI		GNR8_ ACCESS PMI#32 on Current Access: 0 = No 1 = Yes	GNR8_ ACCESS PMI#32 on Next Access: 0 = No 1 = Yes	GNR7_ ACCESS PMI#31 on Current Access: 0 = No 1 = Yes	GNR7_ ACCESS PMI#31 on Next Access: 0 = No 1 = Yes
SYSCFG EAh if AEh[7] = 0			**PMU SMI Source Register 3 (Write 1 to Clear)**				**Default = 00h**
PMI#36, Serial IRQ trap: 0 = Inactive 1 = Active	PMI#35, APM Doze exit: 0 = Inactive 1 = Active	PMI#34, Hot docking time-out SMI: 0 = Inactive 1 = Active	PMI#33, H/W DOZE_ TIMER reload (on Doze exit): 0 = Inactive 1 = Active	PMI#32, GNR4_ ACCESS 0 = Inactive 1 = Active	PMI#31, GNR3_ ACCESS 0 = Inactive 1 = Active	PMI#30, GNR4_ TIMER 0 = Inactive 1 = Active	PMI#29, GNR3_ TIMER 0 = Inactive 1 = Active
SYSCFG EAh if AEh[7] = 1			**PMU SMI Source Register 3A (Write 1 to Clear)**				**Default = 00h**
Reserved				PMI#32, GNR8_ ACCESS 0 = Inactive 1 = Active	PMI#31, GNR7_ ACCESS 0 = Inactive 1 = Active	PMI#30, GNR8_ TIMER 0 = Inactive 1 = Active	PMI#29, GNR7_ TIMER 0 = Inactive 1 = Active
SYSCFG EBh			**Access Port Address Register 2**				**Default = 00h**
Reserved		Access trap address bits A[15:10]: These bits along with SYSCFG D6h[1:0] and D7h[7:0] provide the 16-bit address of the port access that caused the SMI trap. D6h[2] indicates whether an I/O read or an I/O write access was trapped. D6h[3] gives the status of the SBHE# signal for the I/O instruction that was trapped.					
SYSCFG ECh			**Write Trap Register 1 (RO)**				**Default = 00h**
I/O write data trap[15:8]: - Along with SYSCFG EDh[7:0], this register provides the 16-bit write data for trapped I/O write instructions.							
SYSCFG EDh			**Write Trap Register 2 (RO)**				**Default = 00h**
I/O write data trap[7:0]: - Along with SYSCFG ECh[7:0], this register provides the 16-bit write data for trapped I/O write instructions							
SYSCFG EEh			**Power Control Latch Register 4**				**Default = 0Fh**
Enable [3:0] to write latch lines PPWR[15:12]: 0 = Disable 1 = Enable				Read/write data bits for PPWR[15:12] (Default = 1111): 0 = Latch output low 1 = Latch output high			

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG EFh			**Hot Docking Control Register 1**				**Default = 00h**
Hot docking enable: 0 = Disable 1 = Enable (Default)	HDI input debounce rate: 00 = 100μs 01 = 512μs 10 = 1ms 11 = 2ms		HDI active level: 0 = Active high 1 = Active low Also see SYSCFG AAh[0]	HDI SMI: 0 = No SMI on time-out (Default) 1 = Generate SMI on time-out	HDI time-out period: 000 = 1ms 001 = 8ms 010 = 64ms 011 = 256ms 100 = 512ms 101 = 2s 110 = 8s 111 = 16s		
SYSCFG F0h			**Hot Docking Control Register 2**				**Default = 00h**
EPMI3# reload IDLE_TIMER: 0 = Disable 1 = Enable	EPMI2# reload IDLE_TIMER: 0 = Disable 1 = Enable	EPMI1# reload IDLE_TIMER: 0 = Disable 1 = Enable	ROM window feature: 0 = Disable 1 = Enable	ROM window size: 00 = 64KB 01 = 128KB 10 = 256KB 11 = 512KB		EPMI trigger for HDI: 00 = EPMI0# 01 = EPMI1# 10 = EPMI2# 11 = EPMI3# Also see SYSCFG AAh[0]	
SYSCFG F1h			**Low Order Start Address for ROM Window**				**Default = 00h**
Start address bits A[23:19] for ROM window					A18 (for 64KB, 128KB, 256KB window sizes) Ignored for 512KB window size	A17 (for 64KB,128KB window sizes) Ignored for 256KB and 512KB window sizes	A16 (for 64KB window size) Ignored for 128KB, 256KB, and 512KB window sizes
SYSCFG F2h			**High Order Start Address for ROM Window**				**Default = 00h**
Start address bits A[31:24] for ROM window							
SYSCFG F3h			**Thermal Management Register 7**				**Default = 00h**
THFREQ[7:0] - Current frequency low byte: - THFREQ[15:0] return a value from 0 to 65535. This value is updated once per second so that software can read the input pin frequency in kHz and correlate the value to the actual CPU temperature at that moment.							
SYSCFG F4h			**Thermal Management Register 8**				**Default = 00h**
THFREQ[15:8] - Current frequency high byte							
SYSCFG F5h			**PMU Event Register 8**				**Default = 00h**
Reserved		Serial IRQ PMI#36: 00 = Disable 11 = Enable		PCI IRQ driveback trap PMI#38 SMI: 00 = Disable 11 = Enable		DMA_ACCESS PMI#37 SMI: 00 = Disable 11 = Enable	
SYSCFG F5h - FS ACPI Version			**PMU Event Register 8**				**Default = 00h**
ACPI SMI PMI#39: 00 = Disable 11 = Enable		Serial IRQ PMI#36: 00 = Disable 11 = Enable		PCI IRQ driveback trap PMI#38 SMI: 00 = Disable 11 = Enable		DMA_ACCESS PMI#37 SMI: 00 = Disable 11 = Enable	

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG F6h			**DMA Doze Reload Register 1**				**Default = 00h**
DRQ7 reloads DOZE_0: 0 = No 1 = Yes	DRQ6 reloads DOZE_0: 0 = No 1 = Yes	DRQ5 reloads DOZE_0: 0 = No 1 = Yes	IDE DDRQ reloads DOZE_0:(1) 0 = No 1 = Yes	DRQ3 reloads DOZE_0: 0 = No 1 = Yes	DRQ2 reloads DOZE_0: 0 = No 1 = Yes	DRQ1 reloads DOZE_0: 0 = No 1 = Yes	DRQ0 reloads DOZE_0: 0 = No 1 = Yes
(1) Bit 4 controls whether the DDRQ line from bus mastering IDE drives can reload the timers. The bit controls DDRQ from both cables, so enabling the reload feature on any one bus mastering drive enables it for all present.							
SYSCFG F7h			**DMA Doze Reload Register 2**				**Default = 00h**
DRQ7 reloads DOZE_1: 0 = No 1 = Yes	DRQ6 reloads DOZE_1: 0 = No 1 = Yes	DRQ5 reloads DOZE_1: 0 = No 1 = Yes	IDE DDRQ reloads DOZE_1:(1) 0 = No 1 = Yes	DRQ3 reloads DOZE_1: 0 = No 1 = Yes	DRQ2 reloads DOZE_1: 0 = No 1 = Yes	DRQ1 reloads DOZE_1: 0 = No 1 = Yes	DRQ0 reloads DOZE_1: 0 = No 1 = Yes
(1) Bit 4 controls whether the DDRQ line from bus mastering IDE drives can reload the timers. The bit controls DDRQ from both cables, so enabling the reload feature on any one bus mastering drive enables it for all present.							
SYSCFG F8h			**Compact ISA Control Register 1**				**Default = 00h**
Inhibit MRD# and MWR# if SEL# asserted on memory cycle: 0 = No 1 = Yes	Inhibit MRD# and MWR# if SEL# asserted on DMA cycle: 0 = No 1 = Yes	Inhibit IOR# and IOW# if SEL# asserted on I/O cycle: 0 = No 1 = Yes	IRQ15 assignment: 0 = IRQ15 1 = RI	Reserved	Fast CISA memory cycle: 0 = Disable (ISA# = 0) 1 = Enable (ISA# = 1)	Reserved	Compact ISA interface: 0 = Disable 1 = Enable If disabled, can use pins as PIO pins.
SYSCFG F9h			**Compact ISA Control Register 2**				**Default = 00h**
SPKD signal driving: 0 = Always, per AT spec 1 = Sync, per CISA spec	End-of-Interrupt Hold - Delays 8259 recognition of EOI command to prevent false interrupts): 00 = None 01 = 1 ATCLK 10 = 2 ATCLKs 11 = 3 ATCLKs		Stop Clock Count bits - Stop clock cycle indication to CISA devices of how many ATCLKs to expect before the clock will stop: 000 = Reserved 001 = 1 ATCLK (Default) ... 111 = 7 ATCLKs			Generate CISA stop clock cycle (if not already stopped): 00 = Never 01 = On STPCLK# cycles to the CPU (hardware) 10 = Immediately (software) 11 = Reserved	
SYSCFG FAh			**Compact ISA Control Register 3**				**Default = 00h**
CDIR response to IDE cable 1 read 0 = Disable 1 = Enable	CDIR response to IDE cable 0 read 0 = Disable 1 = Enable	Reserved		Resume from Suspend on SEL#/ATB# low: 0 = Disable 1 = Enable	Reserved		Configuration cycle generation: 0 = No action 1 = Run cycle using scratchpad

Table 5-6 SYSCFG 30h-FFh (Power Management) (cont.)

7	6	5	4	3	2	1	0
SYSCFG FBh			**DMA Idle Reload Register**				**Default = 00h**
DRQ7 reloads IDLE_TIMER: 0 = No 1 = Yes	DRQ6 reloads IDLE_TIMER: 0 = No 1 = Yes	DRQ5 reloads IDLE_TIMER: 0 = No 1 = Yes	IDE DDRQ reloads IDLE_TIMER:(1) 0 = No 1 = Yes	DRQ3 reloads IDLE_TIMER: 0 = No 1 = Yes	DRQ2 reloads IDLE_TIMER: 0 = No 1 = Yes	DRQ1 reloads IDLE_TIMER: 0 = No 1 = Yes	DRQ0 reloads IDLE_TIMER: 0 = No 1 = Yes
(1) Bit 4 controls whether the DDRQ line from bus mastering IDE drives can reload the timers. The bit controls DDRQ from both cables, so enabling the reload feature on any one bus mastering drive enables it for all present.							
SYSCFG FCh			**IDE Power Management Assignment Register 1**				**Default = 33h**
IDE Drive 1 I/O access reloads GNR4_TIMER: 0 = No 1 = Yes	IDE Drive 1 I/O access reloads GNR3_TIMER: 0 = No 1 = Yes	IDE Drive 1 I/O access reloads HDU_TIMER: 0 = No 1 = Yes	IDE Drive 1 I/O access reloads DSK_TIMER: 0 = No 1 = Yes	IDE Drive 0 I/O access reloads GNR4_TIMER: 0 = No 1 = Yes	IDE Drive 0 I/O access reloads GNR3_TIMER: 0 = No 1 = Yes	IDE Drive 0 I/O access reloads HDU_TIMER: 0 = No 1 = Yes	IDE Drive 0 I/O access reloads DSK_TIMER: 0 = No 1 = Yes
Note: If a bus mastering drive is used, DDRQ will also reload the enabled timer(s).							
SYSCFG FDh			**IDE Power Management Assignment Register 2**				**Default = 33h**
IDE Drive 3 I/O access reloads GNR4_TIMER: 0 = No 1 = Yes	IDE Drive 3 I/O access reloads GNR3_TIMER: 0 = No 1 = Yes	IDE Drive 3 I/O access reloads HDU_TIMER: 0 = No 1 = Yes	IDE Drive 3 I/O access reloads DSK_TIMER: 0 = No 1 = Yes	IDE Drive 2 I/O access reloads GNR4_TIMER: 0 = No 1 = Yes	IDE Drive 2 I/O access reloads GNR3_TIMER: 0 = No 1 = Yes	IDE Drive 2 I/O access reloads HDU_TIMER: 0 = No 1 = Yes	IDE Drive 2 I/O access reloads DSK_TIMER: 0 = No 1 = Yes
Note: If a bus mastering drive is used, DDRQ will also reload the enabled timer(s).							
SYSCFG FEh			**GPCS# Global Control Register**				**Default = 00h**
GPCS3# decode: 0 = Enable 1 = Disable	GPCS2# decode: 0 = Enable 1 = Disable	GPCS1# decode: 0 = Enable 1 = Disable	GPCS0# decode: 0 = Enable 1 = Disable	GPCS3# read cycles drive XDIR low: 0 = Disable 1 = Enable	GPCS2# read cycles drive XDIR low: 0 = Disable 1 = Enable	GPCS1# read cycles drive XDIR low: 0 = Disable 1 = Enable	GPCS0# read cycles drive XDIR low: 0 = Disable 1 = Enable
SYSCFG FFh			**Reserved**				**Default = 00h**

5.2 PCIDV0 Register Space

The PCI Configuration Register Space designated as PCIDV0 is accessed through Configuration Mechanism #1 as Bus #0, Device #0, and Function #0.

PCIDV0 00h-3Fh are PCI-specific registers while PCIDV0 40h-FFh are system control related registers. Table 5-7 gives the bit formats for these registers.

Table 5-7 PCIDV0 00h-FFh

7	6	5	4	3	2	1	0
PCIDV0 00h			**Vendor Identification Register (RO) - Byte 0**				**Default = 45h**
PCIDV0 01h			**Vendor Identification Register (RO) - Byte 1**				**Default = 10h**
PCIDV0 02h			**Device Identification Register (RO) - Byte 0**				**Default = 01h**
PCIDV0 03h			**Device Identification Register (RO) - Byte 1**				**Default = C7h**
PCIDV0 04h			**Command Register - Byte 0**				**Default = 07h**
Address/data stepping (RO): 0 = Disable (always)	PERR# output pin: 0 = Disable (always)	Reserved	Memory write and invalidate cycle generation (RO): Must = 0 (always) No memory write and invalidate cycles will be generated by the 82C700.	Special cycles (RO): Must = 0 (always) The 82C700 does not respond to the PCI special cycle.	Bus master operations (RO): Must = 1 (always) This allows the 82C700 to perform bus master operations at any time. (Default = 1)	Memory access (RO): Must = 1 (always) The 82C700 allows a PCI bus master access to memory at anytime. (Default = 1)	I/O access (RO): Must = 1 (always) The 82C700 allows a PCI bus master I/O access at any time. (Default = 1)
PCIDV0 05h			**Command Register - Byte 1**				**Default = 00h**
Reserved						Fast back-to-back to different slaves: 0 = Disable 1 = Enable	SERR# output pin (RO): 0 = Disable (always)
PCIDV0 06h			**Status Register - Byte 0**				**Default = 80h**
Fast back-to-back capability (RO): 0 = Not Capable 1 = Capable (Default = 1) Also see PCIDV0 46h[2].	Reserved						
PCIDV0 07h			**Status Register - Byte 1**				**Default = 00h**
Detected parity error: Must = 0 (always)	SERR# status: Must = 0 (always)	Master abort status: Must = 0 (always)	Received target abort status: 0 = No target abort 1 = Target abort occurred	Signaled target abort status: Must = 0 (always)	DEVSEL# timing status (RO): Must = 01 (always) Indicates medium timing selection; the 82C700 asserts the DEVSEL# based on medium timing. (Default = 01)		Data parity detected: Must = 0 (always)

Table 5-7 PCIDV0 00h-FFh (cont.)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">PCIDV0 08h</td><td colspan="4">Revision Identification Register (RO)</td><td colspan="2"><u>Default = 10h</u></td></tr>
<tr><td colspan="8">- The chip revision number in the PCI configuration space consists of two parts: a major revision number and a minor revision number. The 8-bit register is interpreted as x.yh, where for example revision 2.1 of the chip would be read as two BCD digits, 0010 0001b. Software programmers must take care in using the minor revision number, because the number may change between register- and software-equivalent versions of the chip.</td></tr>
<tr><td colspan="2">PCIDV0 09h</td><td colspan="4">Class Code Register (RO) - Byte 0</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2">PCIDV0 0Ah</td><td colspan="4">Class Code Register (RO) - Byte 1</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2">PCIDV0 0Bh</td><td colspan="4">Class Code Register (RO) - Byte 2</td><td colspan="2">Default = 06h</td></tr>
<tr><td colspan="2">PCIDV0 0Ch</td><td colspan="4">Reserved</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2">PCIDV0 0Dh</td><td colspan="4">Master Latency Timer Register (RO)</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2">PCIDV0 0Eh</td><td colspan="4">Header Type Register (RO)</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2">PCIDV0 0Fh</td><td colspan="4">Built-In Self-Test (BIST) Register (RO)</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2">PCIDV0 10h-2Bh</td><td colspan="4">Reserved</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2"><u>PCIDV0 2Ch-2Dh</u></td><td colspan="4"><u>Subsystem Vendor ID</u>
<u>(Write one time only)</u></td><td colspan="2"><u>Default = 00h</u></td></tr>
<tr><td colspan="2"><u>PCIDV0 2Eh-2Fh</u></td><td colspan="4"><u>Subsystem ID</u>
<u>(Write one time only)</u></td><td colspan="2"><u>Default = 00h</u></td></tr>
<tr><td colspan="2">PCIDV0 30h-3Fh</td><td colspan="4">Reserved</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2">PCIDV0 40h</td><td colspan="4">Memory Control Register - Byte 0</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2">PCI video frame buffer write posting hole:
These bits map onto address bits A[31:30]. Together with PCIDV0 41h[7:0] they define the 4MB window where write posting can be masked.</td><td><u>Debug bit, works in conjunction with PCIDV0 42h[7:2].</u>
<u>Intended for use on tester. Not for applications use.</u></td><td>Reserved</td><td>0 = Control of writes being posted on PCI bus is determined by SYSCFG 15h[5:4].
1 = No writes will be posted on PCI bus except writes to video memory and frame buffer areas.
Also see bits 2 and 1.</td><td>Write posting to the video frame buffer control:
If bit 3 = 0:
0 = Enable
1 = Disable
If bit 3 = 1:
0 = Disable
1 = Enable</td><td>Write posting to the video memory (A0000h-BFFFFh) control:
If bit 3 = 0:
0 = Enable
1 = Disable
If bit 3 = 1:
0 = Disable
1 = Enable</td><td>I/O cycle write post control:
0 = Disable
1 = Enable</td></tr>
</table>

Table 5-7 PCIDV0 00h-FFh (cont.)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">PCIDV0 41h</td><td colspan="4">Memory Control Register - Byte 1</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="8">PCI video frame buffer write posting hole:
- These bits map onto address bits A[29:22].
- Together with PCIDV0 40h[7:6] they define the 4MB window where write posting can be masked.</td></tr>
<tr><td colspan="2">PCIDV0 42h</td><td colspan="4">Memory Control Register - Byte 2</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="6">Debug bits, work in conjunction with PCIDV0 40h[5].
Intended for use on tester. Not for applications use.</td><td>This bit must be set to 1 to correct improper operation while switching between DRAM banks.</td><td>Allow HA drive-back during CPU memory accesses:
0 = Disable
1 = Enable</td></tr>
<tr><td colspan="2">PCIDV0 43h</td><td colspan="4">Internal Project Revision - Reserved</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2">PCIDV0 44h</td><td colspan="4">Data Path Register 1</td><td colspan="2">Default = 00h</td></tr>
<tr><td>6DW FIFO for CPU write to PCI:
0 = Disable
1 = Enable</td><td>16QW FIFO for PCI read from DRAM:
0 = Disable
1 = Enable</td><td>16QW FIFO for PCI write to DRAM:
0 = Disable
1 = Enable</td><td>6QW FIFO for CPU write to DRAM:
0 = Disable
1 = Enable</td><td>Memory read accesses in DBC if PCIDV0 44h[0] = 1 and 47h[7] = 1:
0 = SDRAM
1 = Reserved</td><td>DBC ping-pong buffer used for PCI master write X-1-1-1:
0 = Disable
1 = Enable</td><td>DBC ping-pong buffer used for PCI master read X-1-1-1:
0 = Disable
1 = Enable</td><td>Memory read accesses in the DBC:
0 = FP Mode
1 = EDO/ SDRAM</td></tr>
<tr><td colspan="2">PCIDV0 45h</td><td colspan="4">Data Path Control Register 2</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="2">Reserved</td><td>Ping-pong buffer reset of CPU read EDO is qualified with HDOE#.
0 = Disable
1 = Enable</td><td>When IADV# = 0, BE[7:0]# are forced to 00h for write (CPU-to-DRAM and CPU-to-PCI) cycle control.
0 = Disable
1 = Enable</td><td>Reserved</td><td>Byte merge for CPU write to PCI:
0 = Disable
1 = Enable
Not supported.</td><td>Byte merge for CPU write to DRAM:
0 = Disable
1 = Enable</td><td>Reserved</td></tr>
<tr><td colspan="2">PCIDV0 46h</td><td colspan="4">Data Path Control Register 3</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="8">Reserved: Write to 0</td></tr>
<tr><td colspan="2">PCIDV0 47h</td><td colspan="4">Data Path Control Register 4</td><td colspan="2">Default = 00h</td></tr>
<tr><td>SDRAM memory read accesses in DBC:
0 = Disable
1 = Enable</td><td>CPU-to-PCI FIFO clearing when combination changed:
0 = Do not clear
1 = Clear</td><td>PCI-to-DRAM FIFO clearing when combination changed:(1)
0 = Do not clear
1 = Clear</td><td>CPU-to-DRAM FIFO clearing when combination changed:
0 = Do not clear
1 = Clear</td><td>82C700 register is writable:
0 = Enable
1 = Disable (cnfg-writes blocked within DBC)</td><td colspan="3">Reserved</td></tr>
<tr><td colspan="8">(1) Bit 5 must be set to 1 whenever the PCI-to-DRAM FIFO is turned on in the system.</td></tr>
<tr><td colspan="2">PCIDV0 48h</td><td colspan="4">Data Path Control Register 5</td><td colspan="2">Default = 00h</td></tr>
</table>

Table 5-7 PCIDV0 00h-FFh (cont.)

7	6	5	4	3	2	1	0
Reserved		During refresh cycles if this bit = 1, the RAS# corresponding to the bank with size 0 will not be generated for SDRAM.	Reserved		SDRAM mode select 000 = Normal SDRAM mode 001 = NOP command enable 010 = All banks precharge 011 = Mode register command enable 100 = CBR cycle enable All other combinations = Reserved		
PCIDV0 49h-4Bh			**Reserved**				**Default = 00h**
PCIDV0 4Ch			**MCACHE Control Register**				**Default = 00h**
MCACHE: 0 = Disable 1 = Enable Not supported.	Reserved	MCACHE detection (RO): 0 = No MCACHE 1 = MCACHE present Not supported.	Reserved				
PCIDV0 4Dh			**Delay Adjustment Register**				**Default = 00h**
Reserved					Internal DLE0# delay adjustment for DRAM read: 00 = I/O buffer delay 01 = 6ns 10 = 3ns 11 = No delay		Reserved
PCIDV0 4Eh			**SDRAM Control Register**				**Default = 00h**
SDRAM + L2 + pipelining:(1) 0 = Disable 1 = Enable	SDRAM + L2 + pipelining:(1) 0 = Disable 1 = Enable	6 CLK SDRAM leadoff: 0 = Disable 1 = Enable	Reserved				
(1) Bits 7 and 6 have been implemented to solve two problems with the DIRTY signal in systems that have SDRAM + L2 + pipelining.							
PCIDV0 4Fh-FFh			**Reserved**				**Default = 00h**

5.3 PCIDV1 Register Space

The PCI Configuration Register Space designated as PCIDV1 is accessed through Configuration Mechanism #1 as Bus #0, Device #1, and Function #0.

PCIDV1 00h-3Fh are PCI-specific related registers while PCIDV1 40h-FFh are system control related registers. Table 5-8 gives the bit formats for these registers.

Table 5-8 PCIDV1 00h-FFh

7	6	5	4	3	2	1	0
PCIDV1 00h			**Vendor Identification Register (RO) - Byte 0**				**Default = 45h**
PCIDV1 01h			**Vendor Identification Register (RO) - Byte 1**				**Default = 10h**
PCIDV1 02h			**Device Identification Register (RO) - Byte 0**				**Default = 00h**
PCIDV1 03h			**Device Identification Register (RO) - Byte 1**				**Default = C7h**
PCIDV1 04h			**Command Register - Byte 0**				**Default = 07h**
Address/data stepping (RO): 0 = Disable (always)	PERR# output pin: 0 = Disable 1 = Enable	Reserved	Memory write and invalidate cycle generation (RO): Must = 0 (always) No memory write and invalidate cycles will be generated by the 82C700.	Special cycles (R/W): The 82C700 does not respond to the PCI special cycle.	Bus master operations (R/W): This allows the 82C700 to perform bus master operations at any time. (Default = 1)	Memory access (RO): Must = 1 (always) The 82C700 allows a PCI bus master access to memory at anytime. (Default = 1)	I/O access (RO): Must = 1 (always) The 82C700 allows a PCI bus master I/O access at any time. (Default = 1)
PCIDV1 05h			**Command Register - Byte 1**				**Default = 00h**
Reserved						Fast back-to-back to different slaves (RO): 0 = Disable 1 = Enable	SERR# output pin: 0 = Disable 1 = Enable
PCIDV1 06h			**Status Register - Byte 0**				**Default = 80h**
Fast back-to-back capability (RO): 0 = Not Capable 1 = Capable (Default = 1) Also see PCIDV1 46h[2].	Reserved						
PCIDV1 07h			**Status Register - Byte 1**				**Default = 02h**
Parity error detected: 0 = No 1 = Yes	SERR# status (RO): Must = 0 (always)	Master abort status (RO): Must = 0 (always)	Received target abort status (RO): 0 = No target abort 1 = Target abort occurred	Signaled target abort status (RO): Must = 0 (always)	DEVSEL# timing status (RO): Must = 01 (always) Indicates medium timing selection; the 82C700 asserts the DEVSEL# based on medium timing. (Default = 01)		Data parity error detected: 0 = No 1 = Yes

Table 5-8 PCIDV1 00h-FFh (cont.)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">PCIDV1 08h — Revision Identification Register (RO) — <u>Default = 10h</u></td></tr>
<tr><td colspan="8">- The chip revision number in the PCI configuration space consists of two parts: a major revision number and a minor revision number. The 8-bit register is interpreted as x.yh, where for example revision 2.1 of the chip would be read as two BCD digits, 0010 0001b. Software programmers must take care in using the minor revision number, because the number may change between register- and software-equivalent versions of the chip.</td></tr>
<tr><td colspan="8">PCIDV1 09h — Class Code Register (RO) - Byte 0 — Default = 00h</td></tr>
<tr><td colspan="8">PCIDV1 0Ah — Class Code Register (RO) - Byte 1 — Default = 00h</td></tr>
<tr><td colspan="8">PCIDV1 0Bh — Class Code Register (RO) - Byte 2 — Default = 06h</td></tr>
<tr><td colspan="8">PCIDV1 0Ch — Reserved — Default = 00h</td></tr>
<tr><td colspan="8">PCIDV1 0Dh — Master Latency Timer Register (RO) — Default = 00h</td></tr>
<tr><td colspan="8">PCIDV1 0Eh — Header Type Register (RO) — Default = 00h</td></tr>
<tr><td colspan="8">PCIDV1 0Fh — Built-In Self-Test (BIST) Register (RO) — Default = 00h</td></tr>
<tr><td colspan="8">PCIDV1 10h-2Bh — Reserved — Default = 00h</td></tr>
<tr><td colspan="8"><u>PCIDV1 2Ch-2Dh</u> — <u>Subsystem Vendor ID</u> — <u>Default = 00h</u>
<u>(Write one time only)</u></td></tr>
<tr><td colspan="8"><u>PCIDV1 2Eh-2Fh</u> — <u>Subsystem ID</u> — <u>Default = 00h</u>
<u>(Write one time only)</u></td></tr>
<tr><td colspan="8">PCIDV1 30h-40h — Reserved — Default = 00h</td></tr>
<tr><td colspan="8">PCIDV1 41h — Keyboard Controller Select Register — Default = 00h</td></tr>
<tr><td>RDKBDPRT (RO):
Keyboard controller has received Command D0h and has not received the following 060h read.</td><td>WRKBDPRT (RO):
Keyboard controller has received Command D1h and has not received the following 060h write.</td><td>IMMINIT: Generate INIT immediately on FEh Command.
0 = Generate INIT immediately on FEh Command
1 = Wait for halt before INIT for keyboard reset</td><td>KBDEMU: Keyboard emulation
0 = Enable
1 = Disable</td><td><u>KBDCS# includes Port 062h and 066h</u>
<u>0 = Disable</u>
<u>1 = Enable</u></td><td colspan="3">Reserved</td></tr>
<tr><td colspan="8">PCIDV1 42h — Reserved — Default = 00h</td></tr>
</table>

Table 5-8 PCIDV1 00h-FFh (cont.)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">PCIDV1 43h Feature Control Register Default = 00h</td></tr>
<tr><td colspan="2">Buffered DMA (data transfer to/ from DRAM through PCI master):
00 = Original DMA with old protocol
01 = Reserved
10 = Original DMA with PCI master capability
11 = Buffered DMA enable</td><td>Enable DMA or ISA master to preempt PCI master:
0 = Disable
1 = Enable</td><td>Fixed/rotating priority between PCI masters:
0 = Rotating
1 = Fixed</td><td colspan="2">Back-to-back ISA I/O cycle delay:
00 = Delay by 3 ATCLKs
01 = Delay by 12 ATCLKs
10 = No delay
11 = Delay by 12 ATCLKs(1)</td><td>PCI master access to ISA:
0 = Enable
1 = Disable</td><td>ISA bus control signals for memory accesses greater than 16M:
0 = Enable
1 = Disable</td></tr>
<tr><td colspan="8">(1) When bits [3:2] take on the combination of 11, all back-to-back cycles are delayed by 12 AT clocks. This is different from the combinations of 00 and 01 because in the latter case, the delay will be inserted only when an I/O access is followed by a second I/O access with no other type of access occurring in between (e.g., a memory access).</td></tr>
<tr><td colspan="8">PCIDV1 44h-45h Reserved Default = 00h</td></tr>
<tr><td colspan="8">PCIDV1 46h PCI Control Register B - Byte 0 Default = 06h</td></tr>
<tr><td>DMA/ISA access to PCI slave:
0 = Never
1 = When internal LMEM# is not asserted
Master retry always unmasked after 16 PCICLKs.</td><td>XDIR control:
0 = XDIR is controlled for accesses to/from ROM, Kybd controller, RTC
1 = XDIR is controlled only during access to/ from ROM</td><td>Conversion of PERR# to SERR#:
0 = Disable
1 = Enable</td><td>Address parity checking:
0 = Disable
1 = Enable</td><td>Generation of SERR# for target abort:
0 = Disable
1 = Enable</td><td>Fast back-to-back capability:
0 = Disable
1 = Enable
Note: The change on this bit will reflect in PCIDV1 06h[7].</td><td>Subtractive decoding sample point:
0 = Typical sample point
1 = Slow sample point</td><td>Reserved</td></tr>
<tr><td colspan="8">PCIDV1 47h PCI Control Register B - Byte 1 Default = 00h</td></tr>
<tr><td>Write protect ISA bus ROM:
0 = Disable ROMCS# for writes
1 = Enable ROMCS# for writes</td><td>Hidden refresh:
0 = Normal refresh
1 = Hidden refresh
Hidden refresh is not supported - never set this bit to 1.</td><td colspan="2">ATCLK frequency:
00 = PCICLK ÷4
01 = PCICLK ÷3
10 = PCICLK ÷2
11 = PCICLK</td><td>CPU master to PCI slave write: (turnaround between address and data phases)
0 = 1 PCICLK
1 = 0 PCICLK</td><td colspan="3">PCI master to PCI master preemption timer: (preempt after unserviced request pending for X PCICLKs)
000 = No Preemption
001 = 260 PCICLKs
010 = 132 PCICLKs
011 = 68 PCICLKs
100 = 36 PCICLKs
101 = 20 PCICLKs
110 = 12 PCICLKs
111 = 5 PCICLKs</td></tr>
</table>

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 48h			**Strap Option Readback Register - Byte 0**				**Default = 00h**
Reserved	IGERR# strap option selects: 0 = 5.0V ISA 1 = 3.3V ISA	NMI strap option selects: 0 = 5.0V DRAM 1 = 3.3V DRAM	INTR strap option selects: 0 = 5.0V PCI 1 = 3.3V PCI	ROMCS#:KBDCS# strap option selects: 00 = CMD#, ATCLK, BALE 01 = PCICLK3, ATCLK, BALE 10 = PCICLK3, PCICLK4, BALE 11 = PCICLK3, PCICLK4, PCICLK5		RTCWR# strap option selects: 0 = GNT2# 1 = PCICLK2	RTCRD# strap option selects: 0 = GNT1# 1 = PCICLK1
PCIDV1 49h			**Strap Option Readback Register - Byte 1**				**Default = 00h**
Reserved			PCICLK0 strap option selects: 0 = No MCACHE 1 = MCACHE enabled Not supported.	RTCAS strap selects[(1)]	BOFF# strap option selects: 0 = PPWRL 1 = PPWR0#	DBEW# strap option selects: 0 = SA[23:18] pins are SA[23:8] signals 1 = SA[23:18] pins are remapped: SA[23:20] = PPWR[3:0] and SA[19:18] = PPWR[9:8]	A20M# strap selects[(1)]
(1) Bits 3 and 0 work together: 00 = NEC mode & No ISA mode 01 = ISA mode without XD bus 10 = ISA mode with XD bus 11 = No ISA mode							
PCIDV1 4Ah			**ROM Chip Select Register 1**				**Default = 00h**
ROMCS# for F8000h-FFFFFh: 0 = Enable 1 = Disable	ROMCS# for F0000h-F7FFFh: 0 = Enable 1 = Disable	ROMCS# for E8000h-EFFFFh: 0 = Disable 1 = Enable	ROMCS# for E0000h-E7FFFh 0 = Disable 1 = Enable	ROMCS# for D8000h-DFFFFh: 0 = Disable 1 = Enable	ROMCS# for D0000h-D7FFFh: 0 = Disable 1 = Enable	ROMCS# for C8000h-CFFFFh: 0 = Disable 1 = Enable	ROMCS# for C0000h-C7FFFh: 0 = Disable 1 = Enable
PCIDV1 4Bh			**ROM Chip Select Register 2**				**Default = 00h**
ROMCS# for FFFF8000h-FFFFFFFFh: 0 = Enable 1 = Disable	ROMCS# for FFFF0000h-FFFF7FFFh: 0 = Enable 1 = Disable	ROMCS# for FFFE8000h-FFFEFFFFh: 0 = Disable 1 = Enable	ROMCS# for FFFE0000h-FFFE7FFFh: 0 = Disable 1 = Enable	ROMCS# for FFFD8000h-FFFDFFFFh: 0 = Disable 1 = Enable	ROMCS# for FFFD0000h-FFFD7FFFh: 0 = Disable 1 = Enable	ROMCS# for FFFC8000h-FFFCFFFFh: 0 = Disable 1 = Enable	ROMCS# for FFFC0000h-FFFC7FFFh: 0 = Disable 1 = Enable

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 4Ch-4Dh			**Reserved**				**Default = 00h**
PCIDV1 4Eh			**Miscellaneous Control Register 1**				**Default = 00h**
0 = Test 4 Disable 1 = Test 4 Enable Intended for use on tester. Not for applications use.	This bit must be set to 1 to correct improper operation on TC.	Reserved		Pipelined byte merge function: 0 = Disable 1 = Enable	EOP: 0 = Output 1 = Input	Byte merge: 0 = Disable 1 = Enable	ISA master data swap: 0 = Enable 1 = Disable
PCIDV1 4Eh - FS ACPI Version			**Miscellaneous Control Register 1**				**Default = 00h**
0 = Test 4 Disable 1 = Test 4 Enable Intended for use on tester. Not for applications use.	This bit must be set to 1 to correct improper operation on TC.	Always assert MCS16# during buffer DMA ISA master mode: 0 = Disable 1 = Enable Intended for use on tester. Not for applications use.	Allow AD data path when CPU reads ISA slave: 0 = Disable 1 = Enable Intended for use on tester. Not for applications use.	Pipelined byte merge function: 0 = Disable 1 = Enable	EOP: 0 = Output 1 = Input	Byte merge: 0 = Disable 1 = Enable	ISA master data swap: 0 = Enable 1 = Disable

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 4Fh			**Miscellaneous Control Register 2**				**Default = 20h**
Reserved	IDE interface: 0 = Disable 1 = Enable	Primary IDE interface (1F0h): 0 = Disable 1 = Enable (Default)	AT clock wait state control: 0 = Extra WS for ISA cycles 1 = No extra WS	Context Save mode: 0 = Disable 1 = Enable	Timer positive decode in PC98 mode (I/O 77h, 75h, 73h, 71h): 0 = Disable, decode 1 CLK after subtractive decode 1 = Enable, medium decode and remap to 43h, 42h, 41h, and 40h	ROMCS#, KBDCS#, and DBEW# pin selections:[1]	BIOS access after soft reset: 0 = ROM 1 = DRAM
(1) 0 = ROMCS# pin can be ROMCS# or PIO23 if PCIDV1 52h[2] = 0 ROMCS# pin can be ROMCS# and KBDCS# if PCIDV1 52h[2] = 1 KBDCS# pin can be KBDCS# or PIO24 DBEW# pin is DBEW# 1 = ROMCS# pin is ROMCS# and KBDCS# if PCIDV1 52h[2] = 0 or 1 KBDCS# pin is DRD# DBEW# is DWR# **Note:** Also see PCIDV1 52h[2].							
PCIDV1 4Fh - FS ACPI Version			**Miscellaneous Control Register 2**				**Default = 20h**
Allow IRQA, B....H PCI IRQs (when programed as level IRQs) to be shareable with PIO PCI IRQ pins: 0 = Disable 1 = Enable	IDE interface: 0 = Disable 1 = Enable	Primary IDE interface (1F0h): 0 = Disable 1 = Enable (Default)	AT clock wait state control: 0 = Extra WS for ISA cycles 1 = No extra WS	Context Save mode: 0 = Disable 1 = Enable	Timer positive decode in PC98 mode (I/O 77h, 75h, 73h, 71h): 0 = Disable, decode 1 CLK after subtractive decode 1 = Enable, medium decode and remap to 43h, 42h, 41h, and 40h	ROMCS#, KBDCS#, and DBEW# pin selections:[1]	BIOS access after soft reset: 0 = ROM 1 = DRAM
(1) 0 = ROMCS# pin can be ROMCS# or PIO23 if PCIDV1 52h[2] = 0 ROMCS# pin can be ROMCS# and KBDCS# if PCIDV1 52h[2] = 1 KBDCS# pin can be KBDCS# or PIO24 DBEW# pin is DBEW# 1 = ROMCS# pin is ROMCS# and KBDCS# if PCIDV1 52h[2] = 0 or 1 KBDCS# pin is DRD# DBEW# is DWR# **Note:** Also see PCIDV1 52h[2].							

Table 5-8 PCIDV1 00h-FFh (cont.)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">PCIDV1 50h-51h</td><td colspan="4">Reserved</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="3">PCIDV1 52h</td><td colspan="3">Miscellaneous Controller Register 3</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="5">Reserved</td><td>ROMCS# pin selection:
0 = Controlled by PCIDV1 4Fh[1]
1 = ROMCS# pin is always ROMCS# and KBDCS#
Note: Also see PCIDV1 4Fh[1]</td><td>Priority scheme:
0 = Disable
1 = Enable
A setting of 1 will employ a priority scheme that guarantees higher priority for PCI masters during arbitration over DMA and ISA masters for the first 7μs interval after every refresh cycle.</td><td>Concurrent refresh and IDE cycle:
0 = Disable
1 = Enable
ISA devices that rely on accurate refresh addresses for proper operation should disable this bit.</td></tr>
<tr><td colspan="3">PCIDV1 52h - FS ACPI Version</td><td colspan="3">Miscellaneous Controller Register 3</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="3">Reserved</td><td>Abort DDMA remap cycle if claimed by Firestar:
0 = Disable
1 = Enable</td><td>Rsvd pin (A7) function:
0 = Rsvd
1 = SDCKE</td><td>ROMCS# pin selection:
0 = Controlled by PCIDV1 4Fh[1]
1 = ROMCS# pin is always ROMCS# and KBDCS#
Note: Also see PCIDV1 4Fh[1]</td><td>Priority scheme:
0 = Disable
1 = Enable
A setting of 1 will employ a priority scheme that guarantees higher priority for PCI masters during arbitration over DMA and ISA masters for the first 7μs interval after every refresh cycle.</td><td>Concurrent refresh and IDE cycle:
0 = Disable
1 = Enable
ISA devices that rely on accurate refresh addresses for proper operation should disable this bit.</td></tr>
</table>

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 53h			**Miscellaneous Controller Register 4**				**Default = 00h**
SDRAM control on TAG[7:4]: 0 = TAG[7:4] controlled by SYSCFG 00h[5] and SYSCFG 11h[3] 1 = TAG[7:4] become SDRAS#, SDCAS#, DWE#, and SDCKE, respectively **Note:** Setting to 1 will override SYSCFG 11h[3]	MA12 function on either RAS3# or RAS4# pin: 00 = Disable 01 = MA12 on RAS3# pin 10 = MA12 on RAS4# pin 11 = Reserved		SDCKE on IRQSER: 0 = Disable 1 = Enable	SDCKE on SEL/ATB#: 0 = Disable 1 = Enable	MicroChannel support: 0 = Disable, and allows GNT0# pre-emption 1 = Enable (disables Port 000h accesses, tristates AEN), and masks GNT0# pre-emption	Lock flash ROM: 0 = Disable (generates ROMCS# during ROM writing) 1 = Enable (no ROMCS# on ROM writes)	Reserved
PCIDV1 54h			**IRQ Driveback Address Register - Byte 0: Address Bits [7:0]**				**Default = 00h**
IRQ driveback protocol address bits [7:0]: - When an external device logic, such as the 82C824 PC Card Controller or the 82C814 Docking Controller, must generate an interrupt from any source, it follows the IRQ Driveback Protocol and toggles the REQ# line to the 82C700. Once it has the bus, it writes the changed IRQ information to the 32-bit I/O address specified in this register. The 82C700 interrupt controller claims this cycle and latches the new IRQ values. - This register defaults to a value of 00h, which disables the IRQ driveback scheme.							
PCIDV1 55h			**IRQ Driveback Address Register - Byte 1: Address Bits [15:8]**				**Default = 00h**
PCIDV1 56h			**IRQ Driveback Address Register - Byte 2: Address Bits [23:16]**				**Default = 00h**
PCIDV1 57h			**IRQ Driveback Address Register - Byte 3: Address Bits [31:24]**				**Default = 00h**
PCIDV1 58h			**DRQ Remap Base Address Register - Byte 0: Address Bits [7:0]**				**Default = 00h**
DRQ remap base address bits [7:0]: - The distributed DMA protocol requires DMA controller registers for each DMA channel to be individually mapped into I/O space outside the range claimed by ISA devices. Bits A[31:0] of this register specify that base; bits 6:0 are reserved (write 0) because the base address can fall only on 128 byte boundaries. The 82C700 logic uses this base address two ways: 1) to claim accesses to a PCMCIA DMA controller channel; 2) to forward accesses across the bridge to remote devices specified in the DMA Channel Selector Register.							
PCIDV1 59h			**DRQ Remap Base Address Register - Byte 1: Address Bits [15:8]**				**Default = 00h**
PCIDV1 5Ah			**DRQ Remap Base Address Register - Byte 2: Address Bits [23:16]**				**Default = 00h**
PCIDV1 5Bh			**DRQ Remap Base Address Register - Byte 3: Address Bits [31:24]**				**Default = 00h**
PCIDV1 5Ch			**DMA Channel Selector Register**				**Default = 00h**
Ch 7 (DMAC2): 0 = Local 1 = On PCI	Ch 6 (DMAC2): 0 = Local 1 = On PCI	Ch 5 (DMAC2): 0 = Local 1 = On PCI	Hardware Distributed DMA: 0 = Disable 1 = Enable	Ch 3 (DMAC1): 0 = Local 1 = On PCI	Ch 2 (DMAC1): 0 = Local 1 = On PCI	Ch 1 (DMAC1): 0 = Local 1 = On PCI	Ch 0 (DMAC1): 0 = Local 1 = On PCI

Table 5-8 PCIDV1 00h-FFh (cont.)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">PCIDV1 5Dh Reserved Default = 00h</td></tr>
<tr><td colspan="8">PCIDV1 5Eh IRQ Scheme Management Register Default = 00h</td></tr>
<tr><td colspan="2">End-of-Interrupt Holdoff bits [1:0]:
The value of these bits selects the number of retries that will be forced on the PCI bus every time an attempt is made to write I/O Port 020h or 0A0h, where OCW2 of the interrupt controller is set. Multiple retries ensure that a device trying to generate an IRQ driveback will succeed before an EOI command takes effect. This feature eliminates the possibility that an EOI could be registered before a change in IRQ status gets back to the central interrupt controller.</td><td>IRQ driveback data readback selection at PCIDV1 60h-63h:
0 = 1st data phase
1 = 2nd data phase</td><td colspan="5">Reserved</td></tr>
<tr><td colspan="8">PCIDV1 5Fh SYSCFG Base Select Register Default = 00h</td></tr>
<tr><td colspan="8">Configuration Register Index/Data Port Address bits A[15:8]:
- This byte provides the upper address bits of the 16-bit address for the system configuration registers index/data port. Bits A[7:0] always point to 22h/24h. At reset, this register defaults to 0, so the full I/O address for the index/data ports is 0022h/0024h.</td></tr>
<tr><td colspan="8">PCIDV1 60h IRQ Driveback Data Register - Byte 0: Data Bits [7:0] Default = 00h</td></tr>
<tr><td colspan="8">IRQ Driveback Data Bits [7:0]:
- Whenever the 82C700 receives an IRQ driveback cycle, it latches the entire 32-bit data value in this register. If any of the IRQs set active in this driveback are also programmed to generate an SMI (through the standard PMU register settings), SMM code can read this register to determine the exact driveback value written.</td></tr>
<tr><td colspan="8">PCIDV1 61h IRQ Driveback Data Register - Byte 1: Data Bits [15:8] Default = 00h</td></tr>
<tr><td colspan="8">PCIDV1 62h IRQ Driveback Data Register - Byte 2: Data Bits [23:16] Default = 00h</td></tr>
<tr><td colspan="8">PCIDV1 63h IRQ Driveback Data Register - Byte 3: Data Bits [31:24] Default = 00h</td></tr>
</table>

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 64h			**PCI Master Control Register 1**				**Default = 10h**
PCI master write X-1-1-1: 0 = Disable 1 = Enable	PCI master read X-1-1-1: 0 = Disable 1 = Enable	PCI master/IDE concurrence: 0 = Disable 1 = Enable Also see PCIIDE 42h[3]	New AHOLD protocol: 0 = Disable 1 = Enable (Default = 1) (Use HREQ to latch AHOLD)	Non-contiguous byte enables for PCI masters: 0 = Disable 1 = Enable	Reserved		ISA refresh: 0 = Enable 1 = Disable, to increase PCI master bandwidth
PCIDV1 64h - FS ACPI Version			**PCI Master Control Register 1**				**Default = 10h**
PCI master write X-1-1-1: 0 = Disable 1 = Enable	PCI master read X-1-1-1: 0 = Disable 1 = Enable	PCI master/IDE concurrence: 0 = Disable 1 = Enable Also see PCIIDE 42h[3]	New AHOLD protocol: 0 = Disable 1 = Enable (Default = 1) (Use HREQ to latch AHOLD)	Non-contiguous byte enables for PCI masters: 0 = Disable 1 = Enable	Reserved	Synchronize reset for refresh logic (for improved timing): 0 = Enable 1 = Disable	ISA refresh: 0 = Enable 1 = Disable, to increase PCI master bandwidth
PCIDV1 65h			**PCI Master Control Register 2**				**Default = 01h**
Reserved		CPU priority control - When accessing PCI: 00 = CPU is lowest priority 01 = Highest priority after 4 PCI master grants 10 = Highest priority after 2 PCI master grants 11 = Highest priority after 3 PCI master grants		Interrupt request register recover:[1] 0 = Disable 1 = Enable	Select DMA current or base address and counter to be read: 0 = Current 1 = Base	ISA retry for CPU/PCI master access ISA cycle: 0 = Disable 1 = Enable	Use of AHOLD signal during CPU-to-PCI cycles:[2] 0 = Disable 1 = Enable (Default = 1)
(1) This features allows IRR to be accessed at SYSCFG 99h for PIC1 and 9Ah for PIC2. (2) Bit 0 is used only if PCIDV1 64h[4] = 1.							
PCIDV1 65h - FS ACPI Version			**PCI Master Control Register 2**				**Default = 01h**
Reserved	Retry all PCI IDE cycles if buffered DMA occupies the ISA bus: 0 = Enable 1 = Disable	CPU priority control - When accessing PCI: 00 = CPU is lowest priority 01 = Highest priority after 4 PCI master grants 10 = Highest priority after 2 PCI master grants 11 = Highest priority after 3 PCI master grants		Interrupt request register recover:[1] 0 = Disable 1 = Enable	Select DMA current or base address and counter to be read: 0 = Current 1 = Base	ISA retry for CPU/PCI master access ISA cycle: 0 = Disable 1 = Enable	Use of AHOLD signal during CPU-to-PCI cycles:[2] 0 = Disable 1 = Enable (Default = 1)
(1) This features allows IRR to be accessed at SYSCFG 99h for PIC1 and 9Ah for PIC2. (2) Bit 0 is used only if PCIDV1 64h[4] = 1.							

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 66h			**Reserved**				**Default = 00h**
PCIDV1 67h			**Miscellaneous Control Register 5**				**Default = 00h**
PCI arbitration time-out mode: 0 = Disable 1 = Enable See PCIDV1 65h[5:4])	Zero wait state CPU R/W for I/O accesses: 0 = Disable 1 = Enable	Reserved	CPU request for PCI control: 0 = Normal 1 = Reserved	Reserved	Refresh preemption: 0 = Enable 1 = Disable	AHOLD delay: 0 = No delay 1 = Delay AHOLD by 3 PCI CLKs	AD31 in Type 1 configuration cycle: 0: AD31 = 0 1: AD31 = 1
PCIDV1 68h			**PCICLK Control Register 1**				**Default = FFh**
Reserved		PCICLK5: 0 = Disable 1 = Enable	PCICLK4: 0 = Disable 1 = Enable	PCICLK3: 0 = Disable 1 = Enable	PCICLK2: 0 = Disable 1 = Enable	PCICLK1: 0 = Disable 1 = Enable	PCICLK0: 0 = Disable 1 = Enable
PCIDV1 69h			**PCICLK Control Register 2**				**Default = 00h**
Reserved		PCICLK5 affected by CLKRUN#: 0 = No 1 = Yes	PCICLK4 affected by CLKRUN#: 0 = No 1 = Yes	PCICLK3 affected by CLKRUN#: 0 = No 1 = Yes	PCICLK2 affected by CLKRUN#: 0 = No 1 = Yes	PCICLK1 affected by CLKRUN#: 0 = No 1 = Yes	PCICLK0 affected by CLKRUN#: 0 = No 1 = Yes
PCIDV1 6Ah			**PCICLK Skew Adjust Register for PCICLK 0, 1, 2**				**Default = 00h**
Reserved: For PCICLK debug purposes.	Coarse adjustment: 000 = No delay 001 = (PCICLK period ÷2) + ~4ns 010 = (PCICLK period ÷2) + ~8ns 011 = (PCICLK period ÷2) + ~12ns 100 = (PCICLK period ÷2) + ~16ns 101 = (PCICLK period ÷2) + ~20ns 110 = (PCICLK period ÷2) + ~24ns 111 = (PCICLK period ÷2) + ~28ns			Reserved	Fine adjustment: 000 = No delay 001 = Add ~1ns 010 = Add ~2ns 011 = Add ~3ns 100 = Add ~4ns 101 = Add ~5ns 110 = Add ~6ns 111 = Add ~7ns		
Note: If both coarse adjustment and fine adjustment are set to 0 (no delay), PCICLKIN will be routed to PCICLK output with no compensation.							
PCIDV1 6Bh			**PCICLK Skew Adjust Register for PCICLK 3, 4, 5**				**Default = 00h**
Reserved: For PCICLK debug purposes.	Coarse adjustment: 000 = No delay 001 = (PCICLK period ÷2) + ~4ns 010 = (PCICLK period ÷2) + ~8ns 011 = (PCICLK period ÷2) + ~12ns 100 = (PCICLK period ÷2) + ~16ns 101 = (PCICLK period ÷2) + ~20ns 110 = (PCICLK period ÷2) + ~24ns 111 = (PCICLK period ÷2) + ~28ns			Reserved	Fine adjustment: 000 = No delay 001 = Add ~1ns 010 = Add ~2ns 011 = Add ~3ns 100 = Add ~4ns 101 = Add ~5ns 110 = Add ~6ns 111 = Add ~7ns		
Note: If both coarse adjustment and fine adjustment are set to 0 (no delay), PCICLKIN will be routed to PCICLK output with no compensation.							

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 6Ch-6Fh			**Reserved**				**Default = 00h**
PCIDV1 70h			**Leakage Control Register - Byte 0**				**Default = 00h**
W/R#, HITM#, FERR#, SMIACT# Suspend state: 00 = No pull-downs 01 = Pull-down 10 = Reserved 11 = Reserved		BE[7:0]#, M/IO#, D/C#, CACHE#, LOCK# Suspend state: 00 = No pull-downs 01 = Pull-down during BOFF# 10 = Pull-down during Suspend 11 = Pull-down during BOFF# and Suspend		HD[63:0] Suspend and Idle state: 00 = Tristate 01 = Tristate, pull-down 10 = Reserved 11 = Reserved		HA[31:3] Suspend state: 00 = Tristate 01 = Tristate, pull-down 10 = Reserved 11 = Reserved	
PCIDV1 71h			**Leakage Control Register - Byte 1**				**Default = 00h**
IGERR#, A20M# Suspend state: 00 = Drive active 01 = Drive inactive 10 = Tristate, pull-down 11 = Reserved		CPURST, CPUINIT, AHOLD, NMI, INTR, STPCLK# Suspend state: 00 = Drive 01 = Tristate 10 = Reserved 11 = Reserved		BRDY#, NA#, KEN#, EADS#, BOFF#, SMI# Suspend state: 0 = Drive 1 = Tristate	MD[63:0] Suspend state: XX1 = Pull-down at Idle X1X = Pull-down in STPCLK# 1XX = Pull-down in Suspend		
PCIDV1 72h			**Leakage Control Register - Byte 2**				**Default = 00h**
REQ3#, GNT3# Suspend state: 00 = Drive 01 = Tristate 10 = Reserved 11 = Reserved		C/BE[3:0]#, IRDY#, TRDY#, STOP#, AD[31:0], LOCK#, FRAME#, PAR, PERR#, SERR#, DEVSEL#, GNT1#, REQ0#, GNT0#, REQ2#, GNT2# Suspend state: 00 = Drive 01 = Tristate 10 = Tristate, pull-down 11 = Reserved		TAG[7:0] state: X1 = Tristate pull-down in STPCLK# 1X = Tristate pull-down in Suspend		BWE#, GWE# Suspend state: 0 = Drive 1 = Tristate	CACS# Suspend state: 0 = Drive 1 = Tristate, pull-down
PCIDV1 73h			**Leakage Control Register - Byte 3**				**Default = 00h**
CMD# Suspend state: 00 = Drive 01 = Tristate 10 = Tristate, pull-down 11 = Reserved		SD[15:0] Suspend state: 01 = No pull-up/down in Active mode, tristate in Suspend mode 01 = Pull-up in Active mode, tristate in Suspend mode 10 = No pull-up/down in Active mode, pull-down in Suspend mode 11 = Pull-up in Active mode, pull-down in Suspend mode		DBEW# Suspend state: 00 = Drive 01 = Tristate 10 = Tristate, pull-down 11 = Reserved		IRQSER Suspend state: 00 = Drive 01 = Tristate 10 = Tristate, pull-down 11 = Reserved	

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 74h			**Leakage Control Register - Byte 4**				**Default = 00h**
SA[15:0] Suspend state: 00 = Drive 01 = Tristate 10 = Tristate, pull-down 11 = Reserved		SA[23:18] Suspend state: 00 = Drive 01 = Tristate 10 = Tristate, pull-down 11 = Reserved		XD[7:0] Suspend state: 01 = No pull-up/down in Active mode, tristate in Suspend mode 01 = Pull-up in Active mode, tristate in Suspend mode 10 = No pull-up/down in Active mode, pull-down in Suspend mode 11 = Pull-up in Active mode, pull-down in Suspend mode		RFSH#, MRD#, MWR#, IORD#, IOWR# Suspend state: 00 = Drive 01 = Tristate 10 = Tristate, pull-down 11 = Reserved	
PCIDV1 75h			**Leakage Control Register - Byte 5**				**Default = 00h**
Secondary IDE interface in ISA-less mode: 0 = Tristated 1 = Driven	XD bus (primary IDE interface) in no XD bus mode: 0 = Tristated 1 = Driven This bit must be set if the RTCAS:A20M# strap option = 10 to allow IDE control signals to be driven on the XD bus.	MD[63:0] engage pull-down: 0 = Controlled by PCIDV1 71[2:0]h 1 = Pull-down always (overrides PCIDV1 71h[2:0])	TAG[7:0] engage pull-down: 0 = Controlled by PCIDV1 72h[3:2] 1 = Pull-down always (overrides PCIDV1 72h[3:2])	DACK[7:0]# Suspend state: 00 = Drive 01 = Tristate 10 = Tristate 11 = Reserved		RTCAS, RTCRD#, RTCWR# Suspend state: 00 = Drive 01 = Tristate 10 = RTCAS: Tristate, pull-up, RTCRD# and RTCWR#: Tristate, pull-down 11 = Reserved	
PCIDV1 75h - FS ACPI Version			**Leakage Control Register - Byte 5**				**Default = 00h**
Secondary IDE interface in ISA-less mode: 0 = Tristated 1 = Driven	XD bus (primary IDE interface) in no XD bus mode: 0 = Tristated 1 = Driven This bit must be set if the RTCAS:A20M# strap option = 10 to allow IDE control signals to be driven on the XD bus.	MD[63:0] engage pull-down: 0 = Controlled by PCIDV1 71[2:0]h 1 = Pull-down always (overrides PCIDV1 71h[2:0])	TAG[7:0] engage pull-down: 0 = Controlled by PCIDV1 72h[3:2] 1 = Pull-down always (overrides PCIDV1 72h[3:2])	DACK[7:0]# Suspend state: 00 = Drive 01 = Tristate 10 = Drive low 11 = Reserved		RTCAS, RTCRD#, RTCWR# Suspend state: 00 = Drive 01 = Tristate 10 = RTCAS: Tristate, pull-up, RTCRD# and RTCWR#: Tristate, pull-down 11 = Reserved	
PCIDV1 76h			**Hot Docking Leakage Control Register**				**Default = 00h**
Reserved						Hot-docking tristate PCI bus/control: 0 = Disable 1 = Enable	Hot-docking tristate ISA bus/control: 0 = Disable 1 = Enable
PCIDV1 77h-7Fh			**Reserved**				**Default = 00h**

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 80h			**PIO0 Pin (CDOE#) Function Register**				**Default = 00h**
Tristate, pull-down PIO0 during Suspend: 0 = No 1 = Yes	000 = Group 0 (Power Management Inputs) 001 = Group 1 (Power Control Outputs) 010 = Group 2 (Miscellaneous Inputs) 011 = Group 3 (Miscellaneous Outputs) 100 = Group 4 (IDE Controller Outputs) 101 = Group 5 (Gate Logic Inputs) 110 = Group 6 (Logic Outputs) 111 = Group 7 (Reserved)			0000 = Group sub-function 0 0001 = Group sub-function 1 0010 = Group sub-function 2 0011 = Group sub-function 3 0100 = Group sub-function 4 0101 = Group sub-function 5 0110 = Group sub-function 6 0111 = Group sub-function 7		1000 = Group sub-function 8 1001 = Group sub-function 9 1010 = Group sub-function 10 1011 = Group sub-function 11 1100 = Group sub-function 12 1101 = Group sub-function 13 1110 = Group sub-function 14 1111 = Group sub-function 15	
Note: Refer to Section 3.3, Programmable I/O Pins for further information.							
PCIDV1 81h			**PIO1 Pin (TAGWE#) Function Register**				**Default = 00h**
Tristate, pull-down PIO1 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 82h			**PIO2 Pin (ADSC#) Function Register**				**Default = 00h**
Tristate, pull-down PIO2 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 83h			**PIO3 Pin (ADV#) Function Register**				**Default = 00h**
Tristate, pull-down PIO3 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 84h			**PIO4 Pin (RAS2#) Function Register**				**Default = 00h**
Tristate, pull-down PIO4 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 85h			**PIO5 Pin (RAS1#) Function Register**				**Default = 00h**
Tristate, pull-down PIO5 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 86h			**PIO6 Pin (CLKRUN#) Function Register**				**Default = 00h**
Tristate, pull-down PIO6 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 87h			**PIO7 Pin (REQ1#) Function Register**				**Default = 00h**
Tristate, pull-down PIO7 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 88h			**PIO8 Pin (REQ2#) Function Register**				**Default = 00h**
Tristate, pull-down PIO8 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 89h			**PIO9 Pin (DDRQ0) Function Register**				**Default = 00h**
Tristate, pull-down PIO9 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 8Ah			**PIO10 Pin (IRQ1) Function Register**				**Default = 00h**
Tristate, pull-down PIO10 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 8Bh			**PIO11 Pin (IRQ8#) Function Register**				**Default = 00h**
Tristate, pull-down PIO11 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 8Ch	**PIO12 Pin (IRQ12) Function Register**						**Default = 00h**
Tristate, pull-down PIO12 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 8Dh	**PIO13 Pin (IRQ14) Function Register**						**Default = 00h**
Tristate, pull-down PIO13 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 8Eh	**PIO14 Pin (SEL#/ATB#) Function Register**						**Default = 00h**
Tristate, pull-down PIO14 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 8Fh	**PIO15 Pin (RSTDRV) Function Register**						**Default = 00h**
Tristate, pull-down PIO15 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 90h	**PIO16 Pin (SA16) Function Register**						**Default = 00h**
Tristate, pull-down PIO16 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 91h	**PIO17 Pin (SA17) Function Register**						**Default = 00h**
Tristate, pull-down PIO17 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 92h			**PIO18 Pin (IO16#) Function Register**				**Default = 00h**
Tristate, pull-down PIO18 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 93h			**PIO19 Pin (M16#) Function Register**				**Default = 00h**
Tristate, pull-down PIO19 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 94h			**PIO20 Pin (SBHE#) Function Register**				**Default = 00h**
Tristate, pull-down PIO20 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 95h			**PIO21 Pin (SMRD#) Function Register**				**Default = 00h**
Tristate, pull-down PIO21 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 96h			**PIO22 Pin (SMWR#) Function Register**				**Default = 00h**
Tristate, pull-down PIO22 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 97h			**PIO23 Pin (ROMCS#) Function Register**				**Default = 00h**
Tristate, pull-down PIO23 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 98h			**PIO24 Pin (KBDCS#) Function Register**				**Default = 00h**
Tristate, pull-down PIO24 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 99h			**PIO25 Pin (DRQA) Function Register**				**Default = 00h**
Tristate, pull-down PIO25 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 9Ah			**PIO26 Pin (DRQB) Function Register**				**Default = 00h**
Tristate, pull-down PIO26 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 9Bh			**PIO27 Pin (DRQC) Function Register**				**Default = 00h**
Tristate, pull-down PIO27 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 9Ch			**PIO28 Pin (DRQD) Function Register**				**Default = 00h**
Tristate, pull-down PIO28 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 9Dh			**PIO29 Pin (DRQE) Function Register**				**Default = 00h**
Tristate, pull-down PIO29 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 9Eh			**PIO30 Pin (DRQF) Function Register**				**Default = 00h**
Tristate, pull-down PIO30 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 9Fh			**PIO31 Pin (DRQG) Function Register**				**Default = 00h**
Tristate, pull-down PIO31 during Suspend: 0 = No 1 = Yes	Group X selection: Refer to PCIDV1 80h[6:4] for decode.			Group sub-function X selection: Refer to PCIDV1 80h[3:0] for decode.			
PCIDV1 A0h			**Logic Matrix Register 1**				**Default = 00h**
Invert input 01h (whether from PIO pin or from logic matrix output)? 0 = No 1 = Yes	Connect logic input 01h (AND2) to: 000 = PIO pin 001 = Logic 1 010 = Out 2h (AND output) 011 = Out 3h (NAND output) 100 = Out 4h (OR output) 101 = Out 5h (XOR output) 110 = Out 6h (flip-flop 1 output) 111 = Out 7h (flip-flop 2 output)			Invert input 00h (whether from PIO pin or from logic matrix output)? 0 = No 1 = Yes	Connect logic input 00h (AND1) to: Refer to PCIDV1 A0h[6:4] for decode.		
PCIDV1 A1h			**Logic Matrix Register 2**				**Default = 00h**
Invert input 03h? 0 = No 1 = Yes	Connect logic input 03h (NAND) to: Refer to PCIDV1 A0h[6:4] for decode.			Invert input 02h? 0 = No 1 = Yes	Connect logic input 02h (AND3) to: Refer to PCIDV1 A0h[6:4] for decode.		
PCIDV1 A2h			**Logic Matrix Register 3**				**Default = 00h**
Invert input 05h? 0 = No 1 = Yes	Connect logic input 05h (OR2) to: Refer to PCIDV1 A0h[6:4] for decode.			Invert input 04h? 0 = No 1 = Yes	Connect logic input 04h (OR1) to: Refer to PCIDV1 A0h[6:4] for decode.		
PCIDV1 A3h			**Logic Matrix Register 4**				**Default = 00h**
Invert input 07h? 0 = No 1 = Yes	Connect logic input 07h (XOR1) to: Refer to PCIDV1 A0h[6:4] for decode.			Invert input 06h? 0 = No 1 = Yes	Connect logic input 06h (OR3) to: Refer to PCIDV1 A0h[6:4] for decode.		

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 A4h			**Logic Matrix Register 5**				**Default = 00h**
Invert input 09h? 0 = No 1 = Yes	Connect logic input 09h (XOR3) to: Refer to PCIDV1 A0h[6:4] for decode.			Invert input 08h? 0 = No 1 = Yes	Connect logic input 08h (XOR2) to: Refer to PCIDV1 A0h[6:4] for decode.		
PCIDV1 A5h			**Logic Matrix Register 6**				**Default = 00h**
Invert input 0Bh? 0 = No 1 = Yes	Connect logic input 0Bh (flip-flop 1, -D input) to: Refer to PCIDV1 A0h[6:4] for decode.			Invert input 0Ah? 0 = No 1 = Yes	Connect logic input 0Ah (flip-flop 1, PRE# input) to: Refer to PCIDV1 A0h[6:4] for decode.		
PCIDV1 A6h			**Logic Matrix Register 7**				**Default = 00h**
Invert input 0Dh? 0 = No 1 = Yes	Connect logic input 0Dh (flip-flop 1, CLR# input) to: Refer to PCIDV1 A0h[6:4] for decode.			Invert input 0Ch? 0 = No 1 = Yes	Connect logic input 0Ch (flip-flop 1, CPUCLKIN input) to: Refer to PCIDV1 A0h[6:4] for decode.		
PCIDV1 A7h			**Logic Matrix Register 8**				**Default = 00h**
Invert input 0Fh? 0 = No 1 = Yes	Connect logic input 0Fh (flip-flop 2, CPUCLKIN input) to: Refer to PCIDV1 A0h[6:4] for decode.			Invert input 0Eh? 0 = No 1 = Yes	Connect logic input 0Eh (flip-flop 1, D input) to: Refer to PCIDV1 A0h[6:4] for decode.		
PCIDV1 A8h			**PIO Pin Current State Register 1**				**Default = 00h**
Value on PIO7 pin: 0 = Logic low 1 = Logic high	Value on PIO6 pin: 0 = Logic low 1 = Logic high	Value on PIO5 pin: 0 = Logic low 1 = Logic high	Value on PIO4 pin: 0 = Logic low 1 = Logic high	Value on PIO3 pin: 0 = Logic low 1 = Logic high	Value on PIO2 pin: 0 = Logic low 1 = Logic high	Value on PIO1 pin: 0 = Logic low 1 = Logic high	Value on PIO0 pin: 0 = Logic low 1 = Logic high
PCIDV1 A9h			**PIO Pin Current State Register 2**				**Default = 00h**
Value on PIO15 pin: 0 = Logic low 1 = Logic high	Value on PIO14 pin: 0 = Logic low 1 = Logic high	Value on PIO13 pin: 0 = Logic low 1 = Logic high	Value on PIO12 pin: 0 = Logic low 1 = Logic high	Value on PIO11 pin: 0 = Logic low 1 = Logic high	Value on PIO10 pin: 0 = Logic low 1 = Logic high	Value on PIO9 pin: 0 = Logic low 1 = Logic high	Value on PIO8 pin: 0 = Logic low 1 = Logic high
PCIDV1 AAh			**PIO Pin Current State Register 3**				**Default = 00h**
Value on PIO23 pin: 0 = Logic low 1 = Logic high	Value on PIO22 pin: 0 = Logic low 1 = Logic high	Value on PIO21 pin: 0 = Logic low 1 = Logic high	Value on PIO20 pin: 0 = Logic low 1 = Logic high	Value on PIO19 pin: 0 = Logic low 1 = Logic high	Value on PIO18 pin: 0 = Logic low 1 = Logic high	Value on PIO17 pin: 0 = Logic low 1 = Logic high	Value on PIO16 pin: 0 = Logic low 1 = Logic high
PCIDV1 ABh			**PIO Pin Current State Register 4**				**Default = 00h**
Value on PIO31 pin: 0 = Logic low 1 = Logic high	Value on PIO30 pin: 0 = Logic low 1 = Logic high	Value on PIO29 pin: 0 = Logic low 1 = Logic high	Value on PIO28 pin: 0 = Logic low 1 = Logic high	Value on PIO27 pin: 0 = Logic low 1 = Logic high	Value on PIO26 pin: 0 = Logic low 1 = Logic high	Value on PIO25 pin: 0 = Logic low 1 = Logic high	Value on PIO24 pin: 0 = Logic low 1 = Logic high
PCIDV1 ACh-ADh			**Reserved**				**Default = 00h**

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 AEh			**DBE# Select Register 1**				**Default = 01h**
Reserved	DBEX# selection: 000 = Disable (Default) 001 = DBE0#: Cable 0, Drives 0 and 1 010 = DBE0-0#: Cable 0, Drive 0 011 = DBE0-1#: Cable 0, Drive 1 100 = Decode all IDE accesses 101 = DBE1#: Cable 1, Drives 0 and 1 110 = DBE1-0#: Cable 1, Drive 0 111 = DBE1-1#: Cable 1, Drive 1			Reserved	DBEW# selection: 000 = Disable 001 = DBE0#: Cable 0, Drives 0 and 1(Default) 010 = DBE0-0#: Cable 0, Drive 0 011 = DBE0-1#: Cable 0, Drive 1 100 = Decode all IDE accesses 101 = DBE1#: Cable 1, Drives 0 and 1 110 = DBE1-0#: Cable 1, Drive 0 111 = DBE1-1#: Cable 1, Drive 1		
PCIDV1 AFh			**DBE# Select Register 2**				**Default = 00h**
Reserved	DBEZ# selection: 000 = Disable (Default) 001 = DBE0#: Cable 0, Drives 0 and 1 010 = DBE0-0#: Cable 0, Drive 0 011 = DBE0-1#: Cable 0, Drive 1 100 = Decode all IDE accesses 101 = DBE1#: Cable 1, Drives 0 and 1 110 = DBE1-0#: Cable 1, Drive 0 111 = DBE1-1#: Cable 1, Drive 1			Reserved	DBEY# selection: 000 = Disable (Default) 001 = DBE0#: Cable 0, Drives 0 and 1 010 = DBE0-0#: Cable 0, Drive 0 011 = DBE0-1#: Cable 0, Drive 1 100 = Decode all IDE accesses 101 = DBE1#: Cable 1, Drives 0 and 1 110 = DBE1-0#: Cable 1, Drive 0 111 = DBE1-1#: Cable 1, Drive 1		
PCIDV1 B0h			**IRQA Interrupt Selection Register**				**Default = 03h**
Engage pull-down on IRQA? 0 = No 1 = Yes	Reserved		Interrupt source: 0 = ISA (edge) 1 = PCI (level)	Interrupt selection on IRQA pin (Default = IRQ3): 0000 = Disable 0110 = IRQ6 1011 = IRQ11 0001 = IRQ1 0111 = IRQ7 1100 = IRQ12 0010 = Rsvd 1000 = IRQ8# 1101 = Rsvd 0011 = IRQ3 1001 = IRQ9 1110 = IRQ14 0100 = IRQ4 1010 = IRQ10 1111 = IRQ15 0101 = IRQ5			
PCIDV1 B1h			**IRQB Interrupt Selection Register**				**Default = 04h**
Engage pull-down on IRQB? 0 = No 1 = Yes	Reserved		Interrupt source: 0 = ISA (edge) 1 = PCI (level)	Interrupt selection on IRQB pin (Default = IRQ4): 0000 = Disable 0110 = IRQ6 1011 = IRQ11 0001 = IRQ1 0111 = IRQ7 1100 = IRQ12 0010 = Rsvd 1000 = IRQ8# 1101 = Rsvd 0011 = IRQ3 1001 = IRQ9 1110 = IRQ14 0100 = IRQ4 1010 = IRQ10 1111 = IRQ15 0101 = IRQ5			
PCIDV1 B2h			**IRQC Interrupt Selection Register**				**Default = 05h**
Engage pull-down on IRQC? 0 = No 1 = Yes	Reserved		Interrupt source: 0 = ISA (edge) 1 = PCI (level)	Interrupt selection on IRQC pin (Default = IRQ5): 0000 = Disable 0110 = IRQ6 1011 = IRQ11 0001 = IRQ1 0111 = IRQ7 1100 = IRQ12 0010 = Rsvd 1000 = IRQ8# 1101 = Rsvd 0011 = IRQ3 1001 = IRQ9 1110 = IRQ14 0100 = IRQ4 1010 = IRQ10 1111 = IRQ15 0101 = IRQ5			

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 B3h			**IRQD Interrupt Selection Register**				**Default = 06h**
Engage pull-down on IRQD? 0 = No 1 = Yes	Reserved		Interrupt source: 0 = ISA (edge) 1 = PCI (level)	Interrupt selection on IRQD pin (Default = IRQ6): 0000 = Disable 0001 = IRQ1 0010 = Rsvd 0011 = IRQ3 0100 = IRQ4 0101 = IRQ5 0110 = IRQ6 0111 = IRQ7 1000 = IRQ8# 1001 = IRQ9 1010 = IRQ10 1011 = IRQ11 1100 = IRQ12 1101 = Rsvd 1110 = IRQ14 1111 = IRQ15			
PCIDV1 B4h			**IRQE Interrupt Selection Register**				**Default = 07h**
Engage pull-down on IRQE? 0 = No 1 = Yes	Reserved		Interrupt source: 0 = ISA (edge) 1 = PCI (level)	Interrupt selection on IRQE pin (Default = IRQ7): 0000 = Disable 0001 = IRQ1 0010 = Rsvd 0011 = IRQ3 0100 = IRQ4 0101 = IRQ5 0110 = IRQ6 0111 = IRQ7 1000 = IRQ8# 1001 = IRQ9 1010 = IRQ10 1011 = IRQ11 1100 = IRQ12 1101 = Rsvd 1110 = IRQ14 1111 = IRQ15			
PCIDV1 B5h			**IRQF Interrupt Selection Register**				**Default = 09h**
Engage pull-down on IRQF? 0 = No 1 = Yes	Reserved		Interrupt source: 0 = ISA (edge) 1 = PCI (level)	Interrupt selection on IRQF pin (Default = IRQ9): 0000 = Disable 0001 = IRQ1 0010 = Rsvd 0011 = IRQ3 0100 = IRQ4 0101 = IRQ5 0110 = IRQ6 0111 = IRQ7 1000 = IRQ8# 1001 = IRQ9 1010 = IRQ10 1011 = IRQ11 1100 = IRQ12 1101 = Rsvd 1110 = IRQ14 1111 = IRQ15			
PCIDV1 B6h			**IRQG Interrupt Selection Register**				**Default = 0Ah**
Engage pull-down on IRQG? 0 = No 1 = Yes	Reserved		Interrupt source: 0 = ISA (edge) 1 = PCI (level)	Interrupt selection on IRQG pin (Default = IRQ10): 0000 = Disable 0001 = IRQ1 0010 = Rsvd 0011 = IRQ3 0100 = IRQ4 0101 = IRQ5 0110 = IRQ6 0111 = IRQ7 1000 = IRQ8# 1001 = IRQ9 1010 = IRQ10 1011 = IRQ11 1100 = IRQ12 1101 = Rsvd 1110 = IRQ14 1111 = IRQ15			
PCIDV1 B7h			**IRQH Interrupt Selection Register**				**Default = 0Bh**
Engage pull-down on IRQH? 0 = No 1 = Yes	Reserved		Interrupt source: 0 = ISA (edge) 1 = PCI (level)	Interrupt selection on IRQH pin (Default = IRQ11): 0000 = Disable 0001 = IRQ1 0010 = Rsvd 0011 = IRQ3 0100 = IRQ4 0101 = IRQ5 0110 = IRQ6 0111 = IRQ7 1000 = IRQ8# 1001 = IRQ9 1010 = IRQ10 1011 = IRQ11 1100 = IRQ12 1101 = Rsvd 1110 = IRQ14 1111 = IRQ15			

Table 5-8 PCIDV1 00h-FFh (cont.)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">PCIDV1 B8h — PCI Interrupt Selection Register 1 — Default = 00h</td></tr>
<tr><td colspan="4">Interrupt selection on PIO PCIRQ1# input (Default = Disable):
0000 = Disable
0001 = IRQ1
0010 = Rsvd
0011 = IRQ3
0100 = IRQ4
0101 = IRQ5
0110 = IRQ6
0111 = IRQ7
1000 = IRQ8#
1001 = IRQ9
1010 = IRQ10
1011 = IRQ11
1100 = IRQ12
1101 = Rsvd
1110 = IRQ14
1111 = IRQ15</td><td colspan="4">Interrupt selection on PIO PCIRQ0# input (Default = Disable):
0000 = Disable
0001 = IRQ1
0010 = Rsvd
0011 = IRQ3
0100 = IRQ4
0101 = IRQ5
0110 = IRQ6
0111 = IRQ7
1000 = IRQ8#
1001 = IRQ9
1010 = IRQ10
1011 = IRQ11
1100 = IRQ12
1101 = Rsvd
1110 = IRQ14
1111 = IRQ15</td></tr>
<tr><td colspan="8">PCIDV1 B9h — PCI Interrupt Selection Register 2 — Default = 00h</td></tr>
<tr><td colspan="4">Interrupt selection on PIO PCIRQ3# input (Default = Disable):
0000 = Disable
0001 = IRQ1
0010 = Rsvd
0011 = IRQ3
0100 = IRQ4
0101 = IRQ5
0110 = IRQ6
0111 = IRQ7
1000 = IRQ8#
1001 = IRQ9
1010 = IRQ10
1011 = IRQ11
1100 = IRQ12
1101 = Rsvd
1110 = IRQ14
1111 = IRQ15</td><td colspan="4">Interrupt selection on PIO PCIRQ2# input (Default = Disable):
0000 = Disable
0001 = IRQ1
0010 = Rsvd
0011 = IRQ3
0100 = IRQ4
0101 = IRQ5
0110 = IRQ6
0111 = IRQ7
1000 = IRQ8#
1001 = IRQ9
1010 = IRQ10
1011 = IRQ11
1100 = IRQ12
1101 = Rsvd
1110 = IRQ14
1111 = IRQ15</td></tr>
<tr><td colspan="8">PCIDV1 BAh — Serial IRQ Control Register 1 — Default = 00h</td></tr>
<tr><td>Compaq SIRQ HALT mode request:
0 = Active
1 = Halt</td><td>Compaq SIRQ QUIET mode request:
0 = Continuous
1 = Quiet</td><td>SIRQ delays ISR accesses:
0 = No
1 = Yes</td><td>Compaq SIRQ data frame slots. Change only when the serial IRQ logic is disabled or in HALT state:
0 = 17 slots
1 = 21 slots</td><td colspan="2">Compaq SIRQ - Start frame width in PCI clocks. Change this setting only when serial IRQ is disabled or in HALT state:
00 = 4 PCICLKs
01 = 6 PCICLKs
10 = 8 PCICLKs
11 = Reserved</td><td>SIRQ delays EOI accesses:
0 = No
1 = Yes</td><td>Compaq SIRQ (Compaq serial IRQ scheme):
0 = Disable
1 = Enable</td></tr>
<tr><td colspan="8">PCIDV1 BBh — Serial IRQ Control Register 2 — Default = 00h</td></tr>
<tr><td>Compaq SIRQ in HALT state (RO):
0 = No
1 = Yes</td><td>Compaq SIRQ in QUIET state (RO):
0 = No
1 = Yes</td><td colspan="4">Reserved</td><td>SIRQ delays IRR accesses:
0 = No
1 = Yes</td><td>Intel SIRQ (Intel serial IRQ scheme):
0 = Disable
1 = Enable</td></tr>
<tr><td colspan="8">PCIDV1 BCh-BFh — Reserved — Default = 00h</td></tr>
</table>

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 C0h			**DMA Channels A and B Selection Register**				**Default = 10h**
Reserved	DMA channel selection on DRQB/DACKB# pins (Channel 1): 000 = Channel 0 100 = PPWR5 001 = Channel 1 101 = Channel 5 010 = Channel 2 110 = Channel 6 011 = Channel 3 111 = Channel 7			Reserved	DMA channel selection on DRQA/DACKA# pins (Default = Channel 0): 000 = Channel 0 100 = PPWR4 001 = Channel 1 101 = Channel 5 010 = Channel 2 110 = Channel 6 011 = Channel 3 111 = Channel 7		
PCIDV1 C1h			**DMA Channels C and D Selection Register**				**Default = 32h**
Reserved	DMA channel selection on DRQD/DACKD# pins (Channel 3): 000 = Channel 0 100 = PPWR7 001 = Channel 1 101 = Channel 5 010 = Channel 2 110 = Channel 6 011 = Channel 3 111 = Channel 7			Reserved	DMA channel selection on DRQC/DACKC# pins (Channel 2): 000 = Channel 0 100 = PPWR6 001 = Channel 1 101 = Channel 5 010 = Channel 2 110 = Channel 6 011 = Channel 3 111 = Channel 7		
PCIDV1 C2h			**DMA Channel E Selection Register**				**Default = 50h**
Reserved	DMA channel selection on DRQE/DACKE# pins (Default = Channel 5): 000 = Channel 0 100 = PPWR13 001 = Channel 1 101 = Channel 5 010 = Channel 2 110 = Channel 6 011 = Channel 3 111 = Channel 7			Reserved	Function selection on TC pin: 0 = TC 1 = PPWR10	Function selection on AEN pin: 0 = AEN 1 = PPWR11	Function selection on RFSH# pin: 0 = RFSH# 1 = PPWR12
PCIDV1 C3h			**DMA Channels F and G Selection Register**				**Default = 76h**
Reserved	DMA channel selection on DRQG/DACKG# pins (Default = Channel 7): 000 = Channel 0 100 = PPWR15 001 = Channel 1 101 = Channel 5 010 = Channel 2 110 = Channel 6 011 = Channel 3 111 = Channel 7			Reserved	DMA channel selection on DRQF/DACKF# pins (Default = Channel 6): 000 = Channel 0 100 = PPWR14 001 = Channel 1 101 = Channel 5 010 = Channel 2 110 = Channel 6 011 = Channel 3 111 = Channel 7		
PCIDV1 C4h-CFh			**Reserved**				**Default = 00h**

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
Note: PCIDV1 D0h through EEh pertain only to FS ACPI Version. Otherwise they are reserved.							
PCIDV1 D0h - FS ACPI Version		**PM1_BLK Base Address Register - Byte 0: Address Bits [7:0]**					**Default = 00h**
PM1 Block Base Address Bits - Address value A[15:0] defines the 16-bit starting address for PM1_BLK Register Set in system I/O space. The address is required to be paragraph-aligned (on a 16-byte boundary), so bits [3:0] are always 0.							PM1_BLK Register Set: 0 = Disable 1 = Enable
PCIDV1 D1h - FS ACPI Version		**PM1_BLK Base Address Register - Byte 1: Address Bits [15:8]**					**Default = 00h**
PCIDV1 D2h - FS ACPI Version		**PM2_BLK Base Address Register - Byte 0: Address Bits [7:0]**					**Default = 00h**
PM2 Block Base Address Bits - Address value A[15:0] defines the 16-bit starting address for PM2_BLK Register Set in system I/O space. The address is required to be qword-aligned (on an 8-byte boundary), so bits [2:0] are always 0.							PM2_BLK Register Set: 0 = Disable 1 = Enable
PCIDV1 D3h - FS ACPI Version		**PM2_BLK Base Address Register -Byte 1: Address Bits [15:8]**					**Default = 00h**
PCIDV1 D4h - FS ACPI Version		**P_BLK Base Address Register - Byte 0: Address Bits [7:0]**					**Default = 00h**
Processor Block Base Address Bits - Address value A[15:0] defines the 16-bit starting address for P_BLK Register Set in system I/O space. The address is required to be qword-aligned (on an 8-byte boundary), so bits [2:0] are always 0.							P_BLK Register Set: 0 = Disable 1 = Enable
PCIDV1 D5h - FS ACPI Version		**P_BLK Base Address Register - Byte 1: Address Bits [15:8]**					**Default = 00h**
PCIDV1 D6h		**GPE0_BLK Base Address Register - Byte 0: Address Bits [7:0]**					**Default = 00h**
General Purpose Event Block Base Address Bits - Address value A[15:0] defines the 16-bit starting address for GPE0_BLK Register Set in system I/O space. The address is required to be qword-aligned (on an 8-byte boundary), so bits [2:0] are always 0.							GPE0_BLK Register Set: 0 = Disable 1 = Enable
PCIDV1 D7h - FS ACPI Version		**GPE0_BLK Base Address Register - Byte 0: Address Bits [15:8]**					**Default = 00h**
PCIDV1 D8h - FS ACPI Version			**ACPI Source Control Register - Byte 0**				**Default = 00h**
ACPI7 LID: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI6 EC#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI5 USB#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI4 RI#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI3 FRI#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI2 STSCHG#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI1 DOCK#: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI0 UNDOCK#: 0 = IRQ Driveback 1 = Discrete ACPI input
PCIDV1 D9h - FS ACPI Version			**ACPI Source Control Register - Byte 1**				**Default = 00h**
Reserved				ACPI11: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI10: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI9: 0 = IRQ Driveback 1 = Discrete ACPI input	ACPI8: 0 = IRQ Driveback 1 = Discrete ACPI input
Note: The bits in the ACPI Source Control Register (Bytes 0 and 1) select whether the specified ACPI input comes from the IRQ Driveback cycle or from an external pin source (one of the PIO pins or through ACPIMX option).							

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 DAh - FS ACPI Version			**ACPI Source Status Register - Byte 0**				**Default = 00h**
ACPI7 LID: 0 = Low 1 = High	ACPI6 EC#: 0 = Low 1 = High	ACPI5 USB#: 0 = Low 1 = High	ACPI4 RI#: 0 = Low 1 = High	ACPI3 FRI#: 0 = Low 1 = High	ACPI2 STSCHG#: 0 = Low 1 = High	ACPI1 DOCK#: 0 = Low 1 = High	ACPI0 UNDOCK#: 0 = Low 1 = High
PCIDV1 DBh - FS ACPI Version			**ACPI Source Status Register - Byte 1**				**Default = 00h**
Reserved				ACPI11: 0 = Low 1 = High	ACPI10: 0 = Low 1 = High	ACPI9: 0 = Low 1 = High	ACPI8: 0 = Low 1 = High
Note: The bits in the ACPI Source Status Register (Bytes 0 and 1) indicate the current state of the ACPI lines, either the discrete pins or the last IRQ Driveback value depending on the ACPI Source Control Register setting. This information may also be available elsewhere, since the IRQ Driveback values and PIO pin values can be read from other registers. However, this register provides a central means of reading signal state and is especially useful for signals such as LID (which generates an SCI, System Controller Interrupt, on both opening and closing events).							
PCIDV1 DCh - FS ACPI Version			**ACPI Event Resume Control Register - Byte 0**				**Default = 00h**
ACPI7 LID: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI6 EC#: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI5 USB#: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI4 RI#: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI3 FRI#: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI2 STSCHG#: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI1 DOCK#: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI0 UNDOCK#: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend
PCIDV1 DDh - FS ACPI Version			**ACPI Event Resume Control Register - Byte 1**				**Default = 00h**
Reserved				ACPI11: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI10: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI9: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend	ACPI8: 0 = Event will not cause Resume 1 = Event will cause Resume operation if system is in Suspend
Note: The bits in the ACPI Event Resume Control Register (Bytes 0 and 1) select whether the specified ACPI input can wake the system from its Suspend mode. Note that any PCI device that sends its information via the IRQ Driveback cycle will wake the system when it activates its CLKRUN# pin.							
PCIDV1 DEh-DFh			**Reserved**				**Default = 00h**

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 E0h - FS ACPI Version			**SLP_TYP Control Register - Byte 0**				**Default = 00h**
PLVL17: Selects state PPWR17 line will assume when SCTL_ PPWR17 setting is reached. 0 = Low 1 = High	SCTL_PPWR17: Refer to bits [2:0] for decode.			PLVL16: Selects state PPWR16 line will assume when SCTL_ PPWR16 setting is reached. 0 = Low 1 = High	SCTL_PPWR16: 000 = PPWRx switches when SLP_TYP (PM1_BLK Offset 05h[4:2) is set for ACPI S0 system state 001 = PPWRx switches when SLP_TYP (PM1_BLK Offset 05h[4:2) is set for ACPI S0 or S1 system state 010 = PPWRx switches when SLP_TYP (PM1_BLK Offset 05h[4:2) is set for ACPI S0, S1, or S2 system state 011 = PPWRx switches when SLP_TYP (PM1_BLK Offset 05h[4:2) is set for ACPI S0, S1, S2, or S3 system state 100 = PPWRx switches when SLP_TYP (PM1_BLK Offset 05h[4:2) is set for ACPI S0, S1, S2, S3, or S4 system state		
PCIDV1 E1h - FS ACPI Version			**SLP_TYP Control Register - Byte 1**				**Default = 00h**
PLVL19: Selects state PPWR19 line will assume when SCTL_ PPWR19 setting is reached. 0 = Low 1 = High	SCTL_PPWR19: Refer to PCIDV1 E0h[2:0] for decode.			PLVL18: Selects state PPWR18 line will assume when SCTL_ PPWR18 setting is reached. 0 = Low 1 = High	SCTL_PPWR18: Refer to PCIDV1 E0h[2:0] for decode.		
PCIDV1 E2h - FS ACPI Version			**SLP_TYP Control Register - Byte 2**				**Default = 00h**
PLVL21: Selects state PPWR21 line will assume when SCTL_ PPWR21 setting is reached. 0 = Low 1 = High	SCTL_PPWR21: Refer to PCIDV1 E0h[2:0] for decode.			PLVL20: Selects state PPWR20 line will assume when SCTL_ PPWR20 setting is reached. 0 = Low 1 = High	SCTL_PPWR20: Refer to PCIDV1 E0h[2:0] for decode.		
PCIDV1 E3h - FS ACPI Version			**SLP_TYP Control Register - Byte 3**				**Default = 00h**
PLVL23: Selects state PPWR23 line will assume when SCTL_ PPWR23 setting is reached. 0 = Low 1 = High	SCTL_PPWR23: Refer to PCIDV1 E0h[2:0] for decode.			PLVL22: Selects state PPWR22 line will assume when SCTL_ PPWR22 setting is reached. 0 = Low 1 = High	SCTL_PPWR22: Refer to PCIDV1 E0h[2:0] for decode.		

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 E4h - FS ACPI Version			**SLP_TYP Control Register - Byte 4**				**Default = 00h**
PLVL25: Selects state PPWR25 line will assume when SCTL_ PPWR25 setting is reached. 0 = Low 1 = High	SCTL_PPWR25: Refer to PCIDV1 E0h[2:0] for decode.			PLVL24: Selects state PPWR24 line will assume when SCTL_ PPWR24 setting is reached. 0 = Low 1 = High	SCTL_PPWR24: Refer to PCIDV1 E0h[2:0] for decode.		
PCIDV1 E5h - FS ACPI Version			**SLP_TYP Control Register - Byte 5**				**Default = 00h**
PLVL27: Selects state PPWR27 line will assume when SCTL_ PPWR27 setting is reached. 0 = Low 1 = High	SCTL_PPWR27: Refer to PCIDV1 E0h[2:0] for decode.			PLVL26: Selects state PPWR26 line will assume when SCTL_ PPWR26 setting is reached. 0 = Low 1 = High	SCTL_PPWR26: Refer to PCIDV1 E0h[2:0] for decode.		
PCIDV1 E6h - FS ACPI Version			**SLP_TYP Control Register - Byte 6**				**Default = 00h**
PLVL29: Selects state PPWR29 line will assume when SCTL_ PPWR29 setting is reached. 0 = Low 1 = High	SCTL_PPWR29: Refer to PCIDV1 E0h[2:0] for decode.			PLVL28: Selects state PPWR28 line will assume when SCTL_ PPWR28 setting is reached. 0 = Low 1 = High	SCTL_PPWR28: Refer to PCIDV1 E0h[2:0] for decode.		
PCIDV1 E7h - FS ACPI Version			**SLP_TYP Control Register - Byte 7**				**Default = 00h**
PLVL31: Selects state PPWR31 line will assume when SCTL_ PPWR31 setting is reached. 0 = Low 1 = High	SCTL_PPWR31: Refer to PCIDV1 E0h[2:0] for decode.			PLVL30: Selects state PPWR30 line will assume when SCTL_ PPWR30 setting is reached. 0 = Low 1 = High	SCTL_PPWR30: Refer to PCIDV1 E0h[2:0] for decode.		

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 E8h - FS ACPI Version			**Power Control Latch Set Register**				**Default = 00h**
Control line setting: 0 = Low 1 = High	Reserved		PPWR control line to be set: 00000 = PPWR0 00001 = PPWR1 ... ... 11110 = PPWR30 11111 = PPWR31				
PCIDV1 E9h - FS ACPI Version			**Reserved**				**Default = 00h**
PCIDV1 EAh - FS ACPI Version			**Power Control Readback Register - Byte 0**				**Default = FFh**
PPWR7 state: 0 = Low 1 = High	PPWR6 state: 0 = Low 1 = High	PPWR5 state: 0 = Low 1 = High	PPWR4 state: 0 = Low 1 = High	PPWR3 state: 0 = Low 1 = High	PPWR2 state: 0 = Low 1 = High	PPWR1 state: 0 = Low 1 = High	PPWR0 state: 0 = Low 1 = High
PCIDV1 EBh - FS ACPI Version			**Power Control Readback Register - Byte 1**				**Default = FFh**
PPWR15 state: 0 = Low 1 = High	PPWR14 state: 0 = Low 1 = High	PPWR13 state: 0 = Low 1 = High	PPWR12 state: 0 = Low 1 = High	PPWR11 state: 0 = Low 1 = High	PPWR10 state: 0 = Low 1 = High	PPWR9 state: 0 = Low 1 = High	PPWR8 state: 0 = Low 1 = High
PCIDV1 ECh - FS ACPI Version			**Power Control Readback Register - Byte 2**				**Default = F0h**
PPWR23 state: 0 = Low 1 = High	PPWR22 state: 0 = Low 1 = High	PPWR21 state: 0 = Low 1 = High	PPWR20 state: 0 = Low 1 = High	PPWR19 state: 0 = Low 1 = High	PPWR18 state: 0 = Low 1 = High	PPWR17 state: 0 = Low 1 = High	PPWR16 state: 0 = Low 1 = High
PCIDV1 EDh - FS ACPI Version			**Power Control Readback Register - Byte 3**				**Default = F0h**
PPWR31 state: 0 = Low 1 = High	PPWR30 state: 0 = Low 1 = High	PPWR29 state: 0 = Low 1 = High	PPWR28 state: 0 = Low 1 = High	PPWR27 state: 0 = Low 1 = High	PPWR26 state: 0 = Low 1 = High	PPWR25 state: 0 = Low 1 = High	PPWR24 state: 0 = Low 1 = High
PCIDV1 EEh - FS ACPI Version			**ACPI Thermal Control Register**				**Default = 00h**
Reserved		PIO pin FAN control is auto-toggled high: 00 = Never 01 = During Level 1 and 2 STPCLK# modulation 10 = During Level 2 STPCLK# modulation only 11 = Reserved		Temperature event granularity: Selects the bit of the THFREQ value in SYSCFG F3h-F4h that will be monitored such that it generates a thermal management event when it toggles. 0000 = Bit 0 0001 = Bit 1 0010 = Bit 2 0011 = Bit 3 0100 = Bit 4 0101 = Bit 5 0110 = Bit 6 0111 = Bit 7 1000 = Bit 8 1001 = Bit 9 1010 = Bit 10 1011 = Bit 11 1100 = Bit 12 1101 = Bit 13 1110 = Bit 14 1111 = Bit 15			

Table 5-8 PCIDV1 00h-FFh (cont.)

7	6	5	4	3	2	1	0
PCIDV1 EFh-FDh			**Reserved**				**Default = 00h**
PCIDV1 FEh			**Stop Grant Cycle Generation Register (WO)**				**Default = 00h**
Reserved for debug purposes.							
PCIDV1 FFh			**Parity Error Cycle Generation Register**				**Default = 00h**
Reserved for debug purposes.							

5.4 IDE Register Space

5.4.1 IDE Configuration Registers

The configuration space is mapped as Device 14h (AD31 = 1,) Function 0 or as Device 01h (AD12 = 1) Function 1, as controlled by SYSCFG ADh[4]. This section describes the registers implemented in the 256-byte configuration space. All registers not implemented always return zero during read cycles.

Table 5-9 PCIIDE 00h-47h

7	6	5	4	3	2	1	0
PCIIDE 00h			**Vendor ID Register (RO) - Byte 0**				**Default = 45h**
PCIIDE 01h			**Vendor ID Register (RO) - Byte 1**				**Default = 10h**
PCIIDE 02h			**Device ID Register (RO) - Byte 0 (Note)**				**Default = 68h**
PCIIDE 03h			**Device ID Register (RO) - Byte 1 (Note)**				**Default = D5h**
Note: SYSCFG ADh[2] controls the values returned by these registers.							
PCIIDE 04h			**Command Register - Byte 0**				**Default = 45h**
Reserved (RO)	Parity checking: 0 = Parity checking ignored 1 = IDE controller generates PERR# if a parity error occurs during I/O write cycles For I/O read cycles, the IDE controller always generates parity bit.	Reserved (RO)	Memory write and invalidate: 0 = Memory write command 1 = Memory write and invalidate command	Reserved (RO)	IDE controller becomes a PCI master to generate PCI accesses: 0 = Disable 1 = Enable	Reserved	I/O accesses: IDE controller uses this bit to enable/disable I/O accesses. 0 = Disable 1 = Enable (Default = 1)
PCIIDE 05h			**Command Register - Byte 1**				**Default = 00h**
Reserved (RO)							
PCIIDE 06h			**Status Register - Byte 0**				**Default = 80h**
Fast back-to-back transactions (RO): 0 = Disable 1 = Enable (Default = 1)	Reserved (RO)						

Table 5-9 PCIIDE 00h-47h (cont.)

7	6	5	4	3	2	1	0
PCIIDE 07h			**Status Register - Byte 1**				**Default = 02h**
Parity error: This bit is set whenever the IDE controller detects a parity error. It is cleared by writing 80h to this register.	Reserved (RO)	Master abort: As a PCI master, the IDE controller sets this bit to 1 when its transaction is terminated with a master abort.	Target abort: As a PCI master, the IDE controller sets this bit to 1 when its transaction is terminated with a target abort.	Reserved (RO)	Select timing (RO): These read-only bits indicate the allowable timing assertion for DEVSEL#. (Default = 01)		Data parity: As a PCI master, the IDE controller sets this bit to 1 when it detects a data parity error.
PCIIDE 08h			**Revision ID Register (RO)**				**Default = 00h**
PCIIDE 09h			**Class Code Register - Byte 0**				**Default = 80h**
Bus mastering IDE signature (RO): This bit is set to 1 to indicate master mode support. (Default = 1)	Reserved (RO)			Writability of the Native/Legacy bit for Secondary IDE (RO): Determines if bit 2 is RO or R/W. 0 = Bit 2 is RO 1 = Bit 2 is R/W This bit is set only when PCIIDE 40h[3:2] = 11.	Native/Legacy Mode for Secondary IDE: 0 = Legacy 1 = Native	Writability of the Native/ Legacy bit for Primary IDE (RO): Determines if bit 0 is RO or R/W. 0 = Bit 0 is RO 1 = Bit 0 is R/W If PCIIDE 40h[2] = 0, this bit is 0. When PCIIDE 40h[2] = 1, this bit is 1.	Native/Legacy Mode for Primary IDE: 0 = Legacy 1 = Native
PCIIDE 0Ah			**Class Code Register (RO) - Byte 1**				**Default = 01h**
PCIIDE 0Bh			**Class Code Register (RO) - Byte 2**				**Default = 01h**
PCIIDE 0Ch-0Dh			**Reserved**				**Default = 00h**
PCIIDE 0Eh			**Header Type Register (RO)**				**Default = 00h**
Configuration bit for single (default) or multi-function device. - If SYSCFG ADh[4] is set to 1, this register returns 80h denoting a multi-function device.							
PCIIDE 0Fh			**Built-In Self-Test Register (RO)**				**Default = 00h**
PCIIDE 10h-13h			**Primary IDE Command Block Base Address Register**				**Default = 1F1h with PCIIDE 09h[2] = 1 and 40h[2] = 1**
This register is the I/O space indicator for the Drive Command Block. The address block has a size of eight bytes. - Bits [2:0] are read-only and default to 001. - Bits [31:3] are writable if PCIIDE 40h[2] is set to 1. - If PCIIDE 40h[2] = 0, bits [31:0] are read-only and return 0.							

Table 5-9 PCIIDE 00h-47h (cont.)

7	6	5	4	3	2	1	0

PCIIDE 14h-17h — **Primary IDE Control Block Base Address Register** — **Default = 3F5h with PCIIDE 09h[2] = 1 and 40h[2] = 1**

This register is the I/O space indicator for the Drive Control Block. The address block has a size of four bytes.
- Bits [1:0] are read-only and default to 01.
- Bits [31:3] are writable if PCIIDE 40h[2] is set to 1.
- If PCIIDE 40h[2] = 0, bits [31:0] are read-only and return 0.

PCIIDE 18h-1Bh — **Secondary IDE Command Block Base Address Register** — **Default = 171h with PCIIDE 09h[2] =1, 40h[2] = 1, and 40h[3] = 0**

This register is the I/O space indicator for the Drive Command Block. The address block has a size of eight bytes.
- Bits [2:0] are read-only and default to 001.
- Bits [31:3] are writable if PCIIDE 40h[2] is set to 1.
- If PCIIDE 40h[2] = 0, bits [31:0] are read-only and return 0.

PCIIDE 1Ch-1Fh — **Secondary IDE Control Block Base Address Register** — **Default = 375h with PCIIDE 09h[2] = 1, 40h[2] = 1, and 40h[3] = 0**

This register is the I/O space indicator for the Drive Control Block. The address block has a size of four bytes.
- Bits [1:0] are read-only and default to 01.
- Bits [31:3] are writable if PCIIDE 40h[2] is set to 1.
- If PCIIDE 40h[2] = 0, bits [31:0] are read-only and return 0.

PCIIDE 20h-23h — **Bus Master IDE Base Address Register** — **Default = 00000001h**

This register is the I/O base address indicator for the Bus Master IDE Registers. The address block has a size of 16 bytes.
- Bits [3:0] are read-only and default to 0001.
- Bits [31:4] are writable.

PCIIDE 24h-2Bh — **Reserved** — **Default = 00h**

PCIIDE 2Ch-2Dh — **Subsystem Vendor ID** — **Default = 00h**

(Write one time only)

PCIIDE 2Eh-2Fh — **Subsystem ID** — **Default = 00h**

(Write one time only)

PCIIDE 30h-3Ah — **Reserved** — **Default = 00h**

PCIIDE 3Ch — **Interrupt Line Register** — **Default = 00h**

This register indicates which input of the system interrupt controller the IDE interrupt pin is routed to.

PCIIDE 3Dh — **Interrupt Pin Register (RO)** — **Default = FFh**

The content of this register is FFh.

PCIIDE 3Eh-3Fh — **Reserved** — **Default = 00h**

Table 5-9 PCIIDE 00h-47h (cont.)

7	6	5	4	3	2	1	0
PCIIDE 40h			**IDE Initialization Control Register**				**Default = 00h**
Reserved		Enhanced Slave: 0 = 82C621A-compatible mode, uses a 16-byte FIFO in PIO Mode 1 = Enhanced mode, uses a 32-byte FIFO in PIO Mode	Reserved	Secondary IDE: 0 = Enable 1 = Disable This bit is effective only if PCIDV1 4Fh[6] = 1.	Address relocation: Determines if I/O space addresses are relocatable via programming configuration space registers. 0 = Fixed I/O addresses (1F0h-1F7h, 3F6h for primary; 170h-177h, 376h for secondary) 1 = Relocatable I/O addresses	Mode: These bits control the default 16-bit cycle times for all IDE devices and can be overridden by programming the IDE I/O Registers. 00 = $\geq$ 600ns cycle time (PIO Mode 0) 01 = $\geq$ 383ns cycle time (PIO Mode 2) 10 = $\geq$ 240ns cycle time (PIO Mode 1) 11 = $\geq$ 180ns cycle time (PIO Mode 3) These bits are effective only if PCIDV1 4Fh[6] = 1.	
PCIIDE 41h			**Reserved**				**Default = 00h**

Table 5-9 PCIIDE 00h-47h (cont.)

7	6	5	4	3	2	1	0
PCIIDE 42h			**IDE Enhanced Feature Register**				**Default = 00h**
Reserved		Slave IDE FIFO to ISA bus preemption: 0 = Enable 1 = Disable, 82C700 waits to generate ISA cycles until all data in the IDE FIFO is flushed out	IDE arbiter support for PCI/IDE concurrency: 0 = Disable 1 = Enable	PCI memory read line/write and invalidate commands: 0 = Disable 1 = Enable	Concurrent PCI master IDE and IDE cycle: 0 = Disable 1 = Enable (a 48-byte FIFO is used, master IDE to local memory and IDE devices will be accessed concurrently)	X-1-1-1 MIDE PCI master read/write transfers: 0 = Disable 1 = Enable	Reserved
PCIIDE 42h FS ACPI Version			**IDE Enhanced Feature Register**				**Default = 00h**
Reserved	Write concurrency for IDE master cycles: 0 = Disable 1 = Enable	Slave IDE FIFO to ISA bus preemption: 0 = Enable 1 = Disable, 82C700 waits to generate ISA cycles until all data in the IDE FIFO is flushed out	IDE arbiter support for PCI/IDE concurrency: 0 = Disable 1 = Enable	PCI memory read line/write and invalidate commands: 0 = Disable 1 = Enable	Concurrent PCI master IDE and IDE cycle: 0 = Disable 1 = Enable (a 48-byte FIFO is used, master IDE to local memory and IDE devices will be accessed concurrently)	X-1-1-1 MIDE PCI master read/write transfers: 0 = Disable 1 = Enable	Reserved

Table 5-9 PCIIDE 00h-47h (cont.)

7	6	5	4	3	2	1	0
PCIIDE 43h			**IDE Enhanced Mode Register**				**Default = 00h**
Enhanced Mode for Drive 1 on Secondary Channel: Sets 16-bit cycle times for IDE PIO Modes 4 and 5 or Multi-Word DMA Modes 1 and 2. 00 = Disabled, control by corresponding Timing Registers Set 01 = PIO Mode 4 or Multi-Word DMA Mode 1, command inactive for 2 PCICLKs 10 = PIO Mode 5 or Multi-Word DMA Mode 2, command inactive for 1 PCICLK 11 = Reserved Corresponding 170h/171h[3:0] must be set to 0 before these two bits are set to 01 or 10.		Enhanced Mode for Drive 0 on Secondary Channel: Sets 16-bit cycle times for IDE PIO Modes 4 and 5 or Multi-Word DMA Modes 1 and 2. 00 = Disabled, control by corresponding Timing Registers Set 01 = PIO Mode 4 or Multi-Word DMA Mode 1, command inactive for 2 PCICLKs 10 = PIO Mode 5 or Multi-Word DMA Mode 2, command inactive for 1 PCICLK 11 = Reserved Corresponding 170h/171h[3:0] must be set to 0 before these two bits are set to 01 or 10.		Enhanced Mode for Drive 1 on Primary Channel: Sets 16-bit cycle times for IDE PIO Modes 4 and 5 or Multi-Word DMA Modes 1 and 2. 00 = Disabled, control by corresponding Timing Registers Set 01 = PIO Mode 4 or Multi-Word DMA Mode 1, command inactive for 2 PCICLKs 10 = PIO Mode 5 or Multi-Word DMA Mode 2, command inactive for 1 PCICLK 11 = Reserved Corresponding 1F0h/1F1h[3:0] must be set to 0 before these two bits are set to 01 or 10.		Enhanced Mode for Drive 0 on Primary Channel Sets 16-bit cycle times for IDE PIO Modes 4 and 5 or Multi-Word DMA mode 1 and 2. 00 = Disabled, control by corresponding Timing Registers Set 01 = PIO Mode 4 or Multi-Word DMA Mode 1, command inactive for 2 PCICLKs 10 = PIO Mode 5 or Multi-Word DMA Mode 2, command inactive for 1 PCICLK 11 = Reserved Corresponding 1F0h/1F1h[3:0] must be set to 0 before these two bits are set to 01 or 10.	
PCIIDE 44h			**Emulated Bus Master Register**				**Default = 00h**
Reserved				Emulated bus mastering for Cable 1, Drive 1: 0 = Disable 1 = Enable	Emulated bus mastering for Cable 1, Drive 0: 0 = Disable 1 = Enable	Emulated bus mastering for Cable 0, Drive 1: 0 = Disable 1 = Enable	Emulated bus mastering for Cable 0, Drive 0: 0 = Disable 1 = Enable
PCIIDE 44h - FS ACPI Version			**Ultra DMA Configuration Register**				**Default = 00h**
Ultra DMA for Primary channel mode select, Drive 1(1): 00 = Mode 0 01 = Mode 1 10 = Mode 2 11 = Reserved		Ultra DMA for Primary channel mode select, Drive 0(1): 00 = Mode 0 01 = Mode 1 10 = Mode 2 11 = Reserved		Ultra DMA for Secondary channel, Drive 1: 0 = Disable 1 = Enable	Ultra DMA for Secondary channel, Drive 0: 0 = Disable 1 = Enable	Ultra DMA for Primary channel, Drive 1: 0 = Disable 1 = Enable	Ultra DMA for Primary channel, Drive 0: 0 = Disable 1 = Enable
(1) The select bits only effective for CRC setup time.							
PCIIDE 45h			**IDE Interrupt Selection Register**				**Default = 00h**
Secondary Drive 1 interrupt pin: 00 = IRQ10+PCIRQ0# 01 = IRQ11+PCIRQ1# 10 = IRQ14+PCIRQ2# 11 = IRQ15+PCIRQ3#		Secondary Drive 0 interrupt pin: 00 = IRQ10+PCIRQ0# 01 = IRQ11+PCIRQ1# 10 = IRQ14+PCIRQ2# 11 = IRQ15+PCIRQ3#		Primary Drive 1 interrupt pin: 00 = IRQ10+PCIRQ0# 01 = IRQ11+PCIRQ1# 10 = IRQ14+PCIRQ2# 11 = IRQ15+PCIRQ3#		Primary Drive 0 interrupt pin: 00 = IRQ10+PCIRQ0# 01 = IRQ11+PCIRQ1# 10 = IRQ14+PCIRQ2# 11 = IRQ15+PCIRQ3#	
Note: ISA IRQ is selected for Legacy Mode and PCI IRQ is selected for Native Mode (see PCIIDE 09h).							
PCIIDE 45h - FS ACPI Version			**Ultra DMA Configuration Register**				**Default = 00h**
Reserved				Ultra DMA for Secondary channel mode select, Drive 1(1): 00 = Mode 0 01 = Mode 1 10 = Mode 2 11 = Reserved		Ultra DMA for Secondary channel mode select, Drive 0(1): 00 = Mode 0 01 = Mode 1 10 = Mode 2 11 = Reserved	
(1) The select bits only effective for CRC setup time.							

Table 5-9 PCIIDE 00h-47h (cont.)

7	6	5	4	3	2	1	0
PCIIDE 46h - FS ACPI Version			**Emulated IDE Configuration Register**				**Default = 00h**
Fix for I/O 32-bit Mode 4 and Mode 5 timing: 0 = Disable 1 = Enable	Reserved			Emulated bus mastering for Cable 1, Drive 1: 0 = Disable 1 = Enable	Emulated bus mastering for Cable 1, Drive 0: 0 = Disable 1 = Enable	Emulated bus mastering for Cable 0, Drive 1: 0 = Disable 1 = Enable	Emulated bus mastering for Cable 0, Drive 0: 0 = Disable 1 = Enable
PCIIDE 47h - FS ACPI Version			**IDE Interrupt Selection Register**				**Default = FAh**
Secondary Drive 1 interrupt pin: 00 = IRQ10+PCIRQ0# 01 = IRQ11+PCIRQ1# 10 = IRQ14+PCIRQ2# 11 = IRQ15+PCIRQ3#		Secondary Drive 0 interrupt pin: 00 = IRQ10+PCIRQ0# 01 = IRQ11+PCIRQ1# 10 = IRQ14+PCIRQ2# 11 = IRQ15+PCIRQ3#		Primary Drive 1 interrupt pin: 00 = IRQ10+PCIRQ0# 01 = IRQ11+PCIRQ1# 10 = IRQ14+PCIRQ2# 11 = IRQ15+PCIRQ3#		Primary Drive 0 interrupt pin: 00 = IRQ10+PCIRQ0# 01 = IRQ11+PCIRQ1# 10 = IRQ14+PCIRQ2# 11 = IRQ15+PCIRQ3#	
Note: ISA IRQ is selected for Legacy Mode and PCI IRQ is selected for Native Mode (see PCIIDE 09h).							

5.4.2 IDE I/O Registers

5.4.2.1 Primary IDE I/O Registers

The register addresses are referred to in this section by their power-up default addresses If the power-up default is modified by writing to PCIIDE 13h-10h, then these registers will be relocated accordingly.

The IDE controller contains registers at seven I/O ports accessible after two consecutive 16-bit I/O reads from address 1F1h, followed by a byte write 03h to 1F2h. Any other I/O cycle between these two reads will disable access to the IDE controller registers. Refer to Section 4.11.4, Programming Timing Information, for programming details.

Table 5-10 Primary IDE I/O Registers

7	6	5	4	3	2	1	0
I/O Address 1F2h			**Internal ID Register**				**Default = xxh**
Configuration disable (WO): 0 = Enable accesses to internal IDE controller registers 1 = Disable accesses to internal IDE controller registers until another 2 consecutive I/O reads from 1F1h. (Default = 1)	Configuration off (WO): 0 = Enable accesses to internal IDE controller registers 1 = Disable all accesses to internal IDE controller registers until power-down or reset.	Reserved (RO): Write to 0. (Default = xxxx)				Reserved: Must be written 11. If not written to 11, all writes to the IDE I/O Registers will be blocked.	
I/O Address 1F0h			**Read Cycle Timing Register - Timing 0(1)**				**Default = xxh**
Read pulse width: The value programmed in this register plus one determines the DRD# pulse width in PCICLKs (for a 16-bit read from the IDE Data Register). See Table 4-82 or Table 4-83. (Default = xxxx)				Read recovery time: The value programmed in this register plus two determines the recovery time between the end of DRD# and the next DA[2:0]/DCSx# being presented (after a 16-bit read from the IDE Data Register), measured in PCICLKs. See Table 4-82 or Table 4-83. (Default = xxxx)			
(1) Timing 0 can be programmed only if IDE I/O 1F6h[0] = 0. The timing programmed into this register is applied for IDE accesses to drives as selected by 1F3h[3:2] and 1F3h[7].							
I/O Address 1F0h			**Read Cycle Timing Register - Timing 1(1)**				**Default = xxh**
Read pulse width: The value programmed in this register plus one determines the DRD# pulse width in PCICLKs (for a 16-bit read from the IDE Data Register). See Table 4-82 or Table 4-83. (Default = xxxx)				Read recovery time: The value programmed in this register plus two determines the recovery time between the end of DRD# and the next DA[2:0]/DCSx# being presented (after a 16-bit read from the IDE Data Register), measured in PCICLKs. See Table 4-82 or Table 4-83. (Default = xxxx)			
(1) Timing 1 can be programmed only if IDE I/O 1F6h[0] = 1. The timing programmed into this register is applied for IDE accesses to drives as selected by 1F3h[3:2] and 1F3h[7].							

Note: Both Timing 0 and Timing 1 sets share the same address setup and DRDY delay times as programmed in 1F6h[5:4] and 1F6h[3:2].

Table 5-10 Primary IDE I/O Registers (cont.)

7	6	5	4	3	2	1	0
I/O Address 1F1h			**Write Cycle Timing Register - Timing 0[(1)]**				**Default = xxh**
Write pulse width: The value programmed in this register plus one determines the DWR# pulse width in PCICLKs (for a 16-bit write from the IDE Data Register). See Table 4-82 or Table 4-83. (Default = xxxx)				Write recovery time: The value programmed in this register plus two determines the recovery time between the end of DWR# and the next DA[2:0]/DCSx# being presented (after a 16-bit write from the IDE Data Register), measured in PCICLKs. See Table 4-82 or Table 4-83. (Default = xxxx)			
(1) Timing 0 can be programmed only if IDE I/O 1F6h[0] = 0. The timing programmed into this register is applied for IDE accesses to drives as selected by 1F3h[3:2] and 1F3h[7].							
I/O Address 1F1h			**Write Cycle Timing Register - Timing 1[(1)]**				**Default = xxh**
Write pulse width: The value programmed in this register plus one determines the DWR# pulse width in PCICLKs (for a 16-bit write from the IDE Data Register). See Table 4-82 or Table 4-83. (Default = xxxx)				Write recovery time: The value programmed in this register plus two determines the recovery time between the end of DWR# and the next DA[2:0]/DCSx# being presented (after a 16-bit write from the IDE Data Register), measured in PCICLKs. See Table 4-82 or Table 4-83. (Default = xxxx)			
(1) Timing 1 can be programmed only if IDE I/O 1F6h[0] = 1. The timing programmed into this register is applied for IDE accesses to drives as selected by 1F3h[3:2] and 1F3h[7].							
I/O Address 1F3h			**Control Register**				**Default = xxh**
Timing register value select: 0 = Basic 1 = Enhanced	Reserved (RO)		Enable one wait state read: 0 = 2 WS minimum 1 = 1 WS minimum for data reads	Drive 1 timing select: Basic (1F3h[7] = 0): 0 = Determined by PCIIDE 40h[1:0] 1 = Timing 1 Enhanced (1F3h[7] = 1): 0 = Timing 1 1 = Timing 0	Drive 0 timing select: Basic (1F3h[7] = 0): 0 = Determined by PCIIDE 40h[1:0] 1 = Timing 1 Enhanced (1F3h[7] = 1): 0 = Timing 1 1 = Timing 0	Reserved	Reserved (RO): Must be written 1. (Default = 1)
Note: Bits 2, 3 and 7 of the Control Register should be enabled after the Cycle Timing Registers and Miscellaneous Register are programmed. See Table 4-81 for programming options.							
I/O Address 1F5h			**Strap Register**				**Default = xxh**
Reserved (RO): Must be written 1.	Revision number (RO): When the value of this register is set to 11, the contents of REVID Register (08h) should be used to find the revision level of the chip.		DINTR status (RO): Returns the state of the DINTR input.	Mode (RO): Returns information about drive speed as determined by PCIIDE 40h[1:0].		Reserved (RO): Must be written 1. (Default = 1)	PCI CLK speed: 0 = 33MHz 1 = 25MHz

Note: Both Timing 0 and Timing 1 sets share the same address setup and DRDY delay times as programmed in 1F6h[5:4] and 1F6h[3:2].

Table 5-10 Primary IDE I/O Registers (cont.)

7	6	5	4	3	2	1	0
I/O Address 1F6h			**Miscellaneous Register**				**Default = xxh**
Reserved	Read prefetch: 0 = Disable 1 = Enable	Address setup time:(1) The value programmed in this register plus one determines the address setup time between DRD# or DWR# going active and DA[2:0], DCS3#, DCS1# being presented, measured in PCI-CLKs. See Table 4-82 or Table 4-83. (Default = xx)		Delay:(1) The value programmed in this register plus two determines the minimum number of PCICLKs between DRDY# going high and DRD# or DWR# going inactive. See Table 4-82 or Table 4-83. (Default = xxx)			Timing register load select: 0 = Timing 0 (1F0-1F1h accept Timing 0 values) 1 = Timing 1 (1F0-1F1h accept Timing 1 values)
(1) Both Timing 0 and Timing 1 sets have common address setup and DRDY delay times as programmed in 1F6h[5:2].							

5.4.2.2 Secondary IDE I/O Registers

The register addresses are referred to in this section by their power-up default addresses. If the power-up default is modified by writing to PCIIDE 18h-1Bh, then these registers will be relocated accordingly.

The IDE controller contains registers at seven I/O ports accessible after two consecutive 16-bit I/O reads from address 171h, followed by a byte write 03h to 1F2h. Any other I/O cycle between these two reads will disable access to the IDE controller registers. Refer to Section 4.11.4, Programming Timing Information, for more details.

Table 5-11 Secondary IDE I/O Registers

7	6	5	4	3	2	1	0
I/O Address 172h			**Internal ID Register**				**Default = xxh**
Configuration disable: 0 = Enable accesses to internal IDE controller registers 1 = Disable accesses to internal IDE controller registers until another 2 consecutive I/O reads from 171h. (Default = 1)	Configuration off (WO): 0 = Enable accesses to internal IDE controller registers 1 = Disable all accesses to internal IDE controller registers until power-down or reset.	Reserved (RO): Write to 0. (Default = xxxx)				Reserved: Must be written 11. If not, all writes to IDE I/O Registers will be blocked.	
I/O Address 170h			**Read Cycle Timing Register - Timing 0(1)**				**Default = xxh**
Read pulse width: The value programmed in this register plus one determines the DRD# pulse width in PCICLKs (for a 16-bit read from the IDE Data Register). See Table 4-82 or Table 4-83. (Default = xxxx)				Read recovery time: The value programmed in this register plus two determines the recovery time between the end of DRD# and the next DA[2:0]/DCSx# being presented (after a 16-bit read from the IDE Data Register), measured in PCICLKs. See Table 4-82 or Table 4-83. (Default = xxxx)			
(1) Timing 0 can be programmed only if IDE I/O 176h[0] = 0. The timing programmed into this register is applied for IDE accesses to drives as selected by 173h[3:2] and 173h[7].							
I/O Address 170h			**Read Cycle Timing Register - Timing 1(1)**				**Default = xxh**
Read pulse width: The value programmed in this register plus one determines the DRD# pulse width in PCICLKs (for a 16-bit read from the IDE Data Register). See Table 4-82 or Table 4-83. (Default = xxxx)				Read recovery time: The value programmed in this register plus two determines the recovery time between the end of DRD# and the next DA[2:0]/DCSx# being presented (after a 16-bit read from the IDE Data Register), measured in PCICLKs. See Table 4-82 or Table 4-83. (Default = xxxx)			
(1) Timing 1 can be programmed only if IDE I/O 176h[0] = 1. The timing programmed into this register is applied for IDE accesses to drives as selected by 173h[3:2] and 173h[7].							

Table 5-11 Secondary IDE I/O Registers (cont.)

7	6	5	4	3	2	1	0
I/O Address 171h			**Write Cycle Timing Register - Timing 0[1]**				**Default = xxh**
Write pulse width: The value programmed in this register plus one determines the DWR# pulse width in PCICLKs (for a 16-bit write from the IDE Data Register). See Table 4-82 or Table 4-83. (Default = xxxx)				Write recovery time: The value programmed in this register plus two determines the recovery time between the end of DWR# and the next DA[2:0]/DCSx# being presented (after a 16-bit write from the IDE Data Register), measured in PCICLKs. See Table 4-82 or Table 4-83. (Default = xxxx)			
(1) Timing 0 can be programmed only if IDE I/O 176h[0] = 0. The timing programmed into this register is applied for IDE accesses to drives as selected by 173h[3:2] and 173h[7].							
I/O Address 171h			**Write Cycle Timing Register - Timing 1[1]**				**Default = xxh**
Write pulse width: The value programmed in this register plus one determines the DWR# pulse width in PCICLKs (for a 16-bit write from the IDE Data Register). See Table 4-82 or Table 4-83. (Default = xxxx)				Write recovery time: The value programmed in this register plus two determines the recovery time between the end of DWR# and the next DA[2:0]/DCSx# being presented (after a 16-bit write from the IDE Data Register), measured in PCICLKs. See Table 4-82 or Table 4-83. (Default = xxxx)			
(1) Timing 1 can be programmed only if IDE I/O 176h[0] = 1. The timing programmed into this register is applied for IDE accesses to drives as selected by 173h[3:2] and 173h[7].							
I/O Address 173h			**Control Register**				**Default = xxh**
Timing register value select: 0 = Basic 1 = Enhanced	Reserved (RO)			Drive 1 timing select: Basic (173h[7] = 0): 0 = Determined by PCIIDE 40h[1:0] 1 = Timing 1 Enhanced (173h[7] = 1): 0 = Timing 1 1 = Timing 0	Drive 0 timing select: Basic (173h[7] = 0): 0 = Determined by PCIIDE 40h[1:0] 1 = Timing 1 Enhanced (173h[7] = 1): 0 = Timing 1 1 = Timing 0	Reserved	Reserved (RO): Must be written 1. (Default = 1)
Note: Bits 2, 3 and 7 of the Control Register should be enabled after the Cycle Timing Registers and Miscellaneous Register are programmed. See Table 4-81 for programming options.							
I/O Address 175h			**Strap Register**				**Default = xxh**
Reserved: Must be written 1.	Revision number (RO): When the value of this register is set to 11, the contents of REVID Register (08h) should be used to find the revision level of the chip.		SDINTR status (RO): Returns the state of the SDINTR input. (Default = x)	Reserved		Reserved (RO): Must be written 1.	Reserved (RO)

Table 5-11 Secondary IDE I/O Registers (cont.)

7	6	5	4	3	2	1	0
I/O Address 176h			**Miscellaneous Register**				**Default = xxh**
Reserved	Read prefetch: 0 = Disable 1 = Enable	Address setup:[1] The value programmed in this register plus one determines the address setup time between DRD# or DWR# going active and DA[2:0], DCS3#, DCS1# being presented, measured in PCICLKs. See Table 4-82 or Table 4-83. (Default = xx)		DRDY delay:[1] The value programmed in this register plus two determines the minimum number of PCICLKs between DRDY# going high and DRD# or DWR# going inactive. See Table 4-82 or Table 4-83. (Default = xxx)			Timing register load select: 0 = Timing 0 (170-171h accept Timing 0 values) 1 = Timing 1 (170-171h accept Timing 1 values)
(1) Both Timing 0 and Timing 1 sets have common address setup and DRDY delay times as programmed in 1F7h[5:2].							

5.4.3 Bus Master IDE Registers

The bus master IDE function uses 16 bytes of I/O space. The base address of this block of I/O space is pointed to by the Bus Master IDE Base Address Register (PCIIDE 20h-23h). All bus master IDE I/O space registers can be accessed as byte, word, or dword quantities. The description of the 16 bytes of I/O registers is shown in Table 5-12 and the individual bit formats for each register follow in Table 5-13.

Table 5-12 Bus Master IDE Registers

Offset from Base Address	Register Access	Register Name/Function
00h	R/W	Bus Master IDE Command Register for Primary IDE
01h		Device-specific
02h	RWC	Bus Master IDE Status Register for Primary IDE
03h		Device-specific
04h-07h	R/W	Bus Master IDE PRD Table Address for Primary IDE
08h	R/W	Bus Master IDE Command Register for Secondary IDE
09h		Device-specific
0Ah	RWC	Bus Master IDE status Register for Secondary IDE
0Bh		Device-specific
0Ch-0Fh	R/W	Bus Master IDE PRD Table Address for Secondary IDE

Table 5-13 Bus Master IDE Register Formats

7	6	5	4	3	2	1	0
Base Address + 00h		**Bus Master IDE Command Register for Primary IDE**					**Default = 00h**
Reserved				Read or write control: Sets the direction of the bus master transfer. 0 = PCI bus master reads 1 = PCI bus master writes This bit must not be changed when the bus master function is active.	Reserved		Start/Stop bus master: Writing a 1 to this bit enables bus master operation of the controller. Bus master operation begins when this bit is detected changing from 0 to 1. The controller will transfer data between the IDE device and memory only when this bit is set.(1)
(1) Master operation can be halted by writing 0 to this bit. All state information is lost when a 0 is written; master mode operation cannot be stopped and then resumed. If this bit is reset while bus master operation is still active (i.e., the Bus Master IDE Active bit of the Bus Master IDE Status Register for that IDE channel is set) and the drive has not yet finished its data transfer (the Interrupt bit in the Bus Master IDE Status Register for that IDE channel is not set), the bus master command is said to be aborted and data transferred from the drive may be discarded before being written to memory. This bit is intended to be reset after the data transfer is completed, as indicated by either the Bus Master IDE Active bit or the Interrupt bit of the Bus Master IDE Status Register for that IDE channel.							
Base Address + 02h		**Bus Master IDE Status Register for Primary IDE**					**Default = 00h**
Simplex only (RO): This bit indicates that both bus master channels (primary and secondary) can be operated at the same time.	Drive 1 DMA capable: This bit is set by device-dependent code (BIOS or device driver) to indicate that Drive 1 for this channel is capable of DMA transfers, and that the controller has been initialized for optimum performance.	Drive 0 DMA capable: This bit is set by device-dependent code (BIOS or device driver) to indicate that Drive 0 for this channel is capable of DMA transfers, and that the controller has been initialized for optimum performance.	Reserved		Interrupt: This bit is set by the rising edge of the IDE interrupt line. It is cleared when a 1 is written to it by software. Software can use this bit to determine if an IDE device has asserted its interrupt line. When this bit is read as a 1, all data transferred from the drive is visible in system memory.	Error: This bit is set when the IDE controller encounters an error transferring data to/from memory. The exact error condition is bus-specific and can be determined in a bus-specific manner. This bit is cleared when a 1 is written to it by software.	Bus master IDE active: This bit is set when the Start bit is written to the Command Register. It is cleared when the last transfer for a region is performed, where EOT (end of transfer) for that region is set in the region descriptor. It is also cleared when the Start bit is cleared in the Command Register.(1)
(1) When bit 0 is read as 0, all data transferred from the drive during the previous bus master command is visible in system memory, unless the bus master command was aborted.							

Table 5-13 Bus Master IDE Register Formats (cont.)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Base Address + 04h   Descriptor Table Pointer Register for Primary IDE   Default = 00h</td></tr>
<tr><td colspan="8">- Bits [1:0] - Reserved
- Bits [31:2] - Base Address of Descriptor Table: Corresponds to A[31:2].
Note: The Descriptor Table must be dword aligned and must not cross a 64K boundary in memory.</td></tr>
<tr><td colspan="8">Base Address + 08h   Bus Master IDE Command Register for Secondary IDE</td></tr>
<tr><td colspan="4">Reserved</td><td>Read or write control:
This bit sets the direction of the bus master transfer.
0 = PCI bus master reads
1 = PCI bus master writes
This bit must not be changed when the bus master function is active.</td><td colspan="2">Reserved</td><td>Start/Stop bus master:
Writing a 1 to this bit enables bus master operation of the controller. Bus master operation begins when this bit is detected changing from 0 to 1. The controller will transfer data between the IDE device and memory only when this bit is set.(1)</td></tr>
<tr><td colspan="8">(1) Master operation can be halted by writing 0 to this bit. All state information is lost when a 0 is written; master mode operation cannot be stopped and then resumed. If this bit is reset while bus master operation is still active (i.e., the Bus Master IDE Active bit of the Bus Master IDE Status Register for that IDE channel is set) and the drive has not yet finished its data transfer (the Interrupt bit in the Bus Master IDE Status Register for that IDE channel is not set), the bus master command is said to be aborted and data transferred from the drive may be discarded before being written to memory. This bit is intended to be reset after the data transfer is completed, as indicated by either the Bus Master IDE Active bit or the Interrupt bit of the Bus Master IDE Status Register for that IDE channel.</td></tr>
</table>

Table 5-13 Bus Master IDE Register Formats (cont.)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="2">Base Address + 0Ah</td><td colspan="4">Bus Master IDE Status Register for Secondary IDE</td><td colspan="2">Default = 00h</td></tr>
<tr><td>Simplex only (RO):
This bit indicates that both bus master channels (primary and secondary) can be operated at the same time.</td><td>Drive 1 DMA Capable:
This bit is set by device dependent code (BIOS or device driver) to indicate that Drive 1 for this channel is capable of DMA transfers, and that the controller has been initialized for optimum performance.</td><td>Drive 0 DMA Capable:
This bit is set by device dependent code (BIOS or device driver) to indicate that Drive 0 for this channel is capable of DMA transfers and that the controller has been initialized for optimum performance.</td><td colspan="2">Reserved</td><td>Interrupt:
This bit is set by the rising edge of the IDE interrupt line. It is cleared when a 1 is written to it by software.
Software can use this bit to determine if an IDE device has asserted its interrupt line. When this bit is read as a 1, all data transferred from the drive is visible in system memory.</td><td>Error:
This bit is set when the controller encounters an error transferring data to/from memory. The exact error condition is bus-specific and can be determined in a bus-specific manner.
This bit is cleared when a 1 is written to it by software.</td><td>Bus master IDE active:
This bit is set when the Start bit is written to the Command Register. It is cleared when the last transfer for a region is performed, where EOT (end of transfer) for that region is set in the region descriptor. It is also cleared when the Start bit is cleared in the Command Register.(1)</td></tr>
<tr><td colspan="8">(1) When bit 0 is read as 0, all data transferred from the drive during the previous bus master command is visible in system memory, unless the bus master command was aborted.</td></tr>
<tr><td colspan="2">Base Address + 0Ch</td><td colspan="4">Descriptor Table Pointer Register for Secondary IDE</td><td colspan="2">Default = 00h</td></tr>
<tr><td colspan="8">- Bits [1:0] - Reserved
- Bits [31:2] - Base Address of Descriptor Table: Corresponds to A[31:2].
Note: The Descriptor Table must be dword aligned and must not cross a 64K boundary in memory.</td></tr>
</table>

5.5 I/O Register Space

5.5.1 ISA-Compatible I/O Registers

Table 5-14 is a register map that includes system control registers that are present in FireStar. These registers are directly accessible (CPU direct I/O R/W) in System I/O Register Space.

Table 5-14 ISA-Compatible I/O Register Map

Port	Name	Comment
DMAC1 Control Registers		
000h	Memory Address Register for DMA Channel 0	
001h	Count Register for DMA Channel 0	
002h	Memory Address Register for DMA Channel 1	
003h	Count Register for DMA Channel 1	
004h	Memory Address Register for DMA Channel 2	
005h	Count Register for DMA Channel 2	
006h	Memory Address Register for DMA Channel 3	
007h	Count Register for DMA Channel 3	
008h (Read)	Status Register	
008h (Write)	Command Register	
009h	Request Register	
00Ah (Read)	Command Register	
00Ah (Write)	Set Single Mask Bits	
00Bh	Mode Register	Read 00Eh, then read 00Bh four times to get the Mode Register values.
00Ch (Read)	Set Byte Pointer Flip-Flop	
00Ch (Write)	Clear Byte Pointer Flip-Flop	
00Dh (Red)	Temporary Register	
00Dh (Write)	Master Clear	
00Eh (Read)	Reset Mode Register Read-Back Counter	
00Eh (Write)	Clear Mask	
00Fh	Mask Register	
010h-01Fh	Reserved	
INTC1 Control Registers		
020h	Control Register (see text)	
021h	Control Register (see text)	
Chipset Configuration Registers		
022h	Integrated 82C206 and Chipset Configuration Index Register (SYSCFG)	
023h	Integrated 82C206 Configuration Data Register	
024h	Chipset Configuration Data Register (SYSCFG)	
025h-03Fh	Reserved	
Timer Registers		
040h	Timer Channel 0 Register	
041h	Timer Channel 1 Register	
042h	Timer Channel 2 Register	

Table 5-14 ISA-Compatible I/O Register Map (cont.)

Port	Name	Comment
043h	Timer Control Register	
045h-05Fh	Reserved	
Keyboard Controller Registers		
060h	Reserved for External Keyboard Controller	Access monitored for fast A20M#/RESET
061h	System Control Port B	
062h-063h	Reserved	
064h	Reserved for External Keyboard Controller	Access monitored for fast A20M#/RESET
065h-06Fh	Reserved	
070h	RTC Index Register	Access monitored for RTC control generation and NMI enabling.
071h	RTC Data Register	Access monitored for RTC control generation.
072h-080h	Reserved	
DMA Page Registers		
081h	Page Address Register for DMA Channel 2	
082h	Page Address Register for DMA Channel 3	
083h	Page Address Register for DMA Channel 1	
084h-086h	Reserved	
087h	Page Address Register for DMA Channel 0	
088h	Reserved	
089h	Page Address Register for DMA Channel 6	
08Ah	Page Address Register for DMA Channel 7	
08Bh	Page Address Register for DMA Channel 5	
08Ch-08Eh	Reserved	
08Fh	Page Address Register for DMA Channel 4	
090h-091h	Reserved	
092h	System Control Port A	
093h-09Fh	Reserved	
INTC2 Control Registers		
0A0h	Control Register (see text)	
0A1h	Control Register (see text)	
0A2h-0BFh	Reserved	
DMAC2 Control Registers		
0C0h	Memory Address Register for DMA Channel 4	
0C1h	Reserved	
0C2h	Count Register for DMA Channel 4	
0C3h	Reserved	
0C4h	Memory Address Register for DMA Channel 5	
0C5h	Reserved	
0C6h	Count Register for DMA Channel 5	
0C7h	Reserved	
0C8h	Memory Address Register for DMA Channel 6	
0C9h	Reserved	

Table 5-14 ISA-Compatible I/O Register Map (cont.)

Port	Name	Comment
0CAh	Count Register for DMA Channel 6	
0CBh	Reserved	
0CCh	Memory Address Register for DMA Channel 7	
0CDh	Reserved	
0CEh	Count Register for DMA Channel 7	
0CFh	Reserved	
0D0h (Read)	Status Register	
0D0h (Write)	Command Register	
0D1h	Reserved	
0D2h	Request Register	
0D3h	Reserved	
0D4h (Read)	Command Register	
0D4h (Write)	Set Single Mask Bits	
0D5h	Reserved	
0D6h	Mode Register	Read 0DCh, then read 0D6h four times to get the Mode Register values.
0D7h	Reserved	
0D8h (Read)	Set Byte Pointer Flip-Flop	
0D8h (Write)	Clear Byte Pointer Flip-Flop	
0D9h	Reserved	
0DAh (Red)	Temporary Register	
0DAh (Write)	Master Clear	
0DBh	Reserved	
0DCh (Read)	Reset Mode Register Read-back Counter	
0DCh (Write)	Clear Mask	
0DDh	Reserved	
0DEh	Mask Register	
0DFh	Reserved	
0E0h-0FFh	Reserved	
100h-40Ah	Reserved	
40Bh	EISA DMA Extended Mode Register	
40Ch-4D5h	Reserved	
4D6h	EISA DMA Extended Mode Register	
4D7h-CF7h	Reserved	
CF8h-CFBh	PCI Configuration Index Register (PCIDV0-1)	
CFCh-CFFh	PCI Configuration Data Register (PCIDV0-1)	
D00h-FFFh	Reserved	

5.5.2 ACPI I/O Registers

Tables 5-15 through 5-18 are register maps that include ACPI control and status registers that are present in the ACPI logic module. These registers are directly accessible (CPU direct I/O R/W) in System I/O Register Space, once the corresponding base addresses have been programmed through PCIDV1 D0h-D7h.

Table 5-15 Offset from PM1_BLK Base Address (PCIDV1 D0h-D1h)

<table>
<tr><th>7</th><th>6</th><th>5</th><th>4</th><th>3</th><th>2</th><th>1</th><th>0</th></tr>
<tr><td colspan="8">Offset 00h</td></tr>
<tr><td colspan="2">Reserved</td><td>GBL_STS
Global service status:
Has software written BIOS_RLS = 1?
0 = No
1 = Yes
Write 1 to clear</td><td>BM_STS
Bus master monitor status:
Has any REQ# gone active since this bit was last cleared?
0 = No
1 = Yes
Write 1 to clear</td><td colspan="3">Reserved</td><td>TMR_STS:
Timer status:
Has TMR_VAL[23] toggled (changed from high-to-low or low-to-high)?
0 = No
1 = Yes
Write 1 to clear</td></tr>
<tr><td colspan="8">Offset 01h</td></tr>
<tr><td>WAK_STS
Wakeup status:
Did system wake from Suspend mode after an enabled Resume event occurred?
0 = No
1 =Yes</td><td colspan="3">Reserved</td><td>PWRBTN OR_STS
Power button override status:
PWRBTN# asserted for > 4 sec?
0 = No
1 = Yes</td><td>RTC_STS
RTC status:
Has IRQ8# from RTC gone active?
0 = No
1 = Yes</td><td>Reserved</td><td>PWRBTN_STS
Power button status:
Has user pressed power button?
0 = No
1 = Yes</td></tr>
<tr><td colspan="8">Offset 02h</td></tr>
<tr><td colspan="2">Reserved</td><td>GBL_EN
Global service enable:
Should GBL_STS going to 1 cause SCI?
0 = No
1 = Yes</td><td colspan="4">Reserved</td><td>TMR_EN
Timer enable:
Should TMR_STS going to 1 cause SCI?
0 = No
1 = Yes</td></tr>
<tr><td colspan="8">Offset 03h</td></tr>
<tr><td colspan="5">Reserved</td><td>RTC_EN
RTC enable:
Should RTC_STS going to 1 cause SCI?
0 = No
1 = Yes</td><td>Reserved</td><td>PWRBTN_EN
Power button enable:
Should PWRBTN_STS going to 1 cause SCI?
0 = No
1 = Yes</td></tr>
</table>

Table 5-15 Offset from PM1_BLK Base Address (PCIDV1 D0h-D1h) (cont.)

7	6	5	4	3	2	1	0
Offset 04h							
Reserved					GBL_RLS Global service lock release: Does ACPI software wish to generate SMI to BIOS? 0 = No 1 = Yes	BM_RLD: Bus master monitor RLD: Should BM_STS going to 1 wake up CPU (state restored to C0 from C3)? 0 = No 1 = Yes	SCI_EN System controller interrupt enable: If SCI occurs, generate: 0 = SMI 1 = IRQ13
Offset 05h							
Reserved		SLP_EN Sleep enable: When written to 1, forces SLP_TYP Suspend mode. Always reads 0.	SLP_TYP Sleep mode type: Defines sleep mode to enter when software sets SLP_EN = 1. ACPI ROM table associates 3-bit binary values with one of the system states S0-S4. 000 = S0: Active mode. Clock throttling, etc. determined by CPU state C0-C3. 001 = S1: Low-power Suspend mode with CPU and L2 cache alive. 010 = S2: Same as S1, but power is removed from CPU, L2 cache, and selected peripheral devices. 011 = S3: Same as S2, but power is removed from more devices. 100 = S4: Same as S3, but power is also removed from DRAM in this mode.			Reserved	
Offset 06h-07h			**Reserved**				
Offset 08h-0Ah			**TMR_VAL - Timer Value Register (RO)**				
Bits [23:0] correspond to: [7:0] = 08h, [15:8] = 09h, [23:16] = 0Ah The timer is a free-running “up” counter based on the 14MHz clock divided by 4. It runs whenever the 14MHz input clock to FireStar is present, and is cleared to 0 whenever PCIRST# is asserted. Whenever TMR_VAL[23] changes from 0-to-1 or from 1-to-0, the TMR_STS bit is set to 1; writing 1 back to TMR_STS clears the bit. If TMR_EN = 1 when TMR_STS = 1, an SCI occurs (if globally enabled). - Bits [23:0] = A read-only value that returns the power management timer count. The count is based on 14.31818MHz/4. The count is cleared by a PCI bus reset. Whenever bit 23 toggles, TMR_STS is set to indicate the event. Counts only while the system is active.							
Offset 0Bh-0Dh			**Reserved**				

Table 5-16 Offset from PM2_BLK Base Address (PCIDV1 D2h-D3h)

7	6	5	4	3	2	1	0
Offset 00h							
Reserved							ARB_DIS Arbitration disable: Software uses this bit to enable and disable system master devices. 0 = Enable arbitration 1 = Disable arbitration
Offset 01h-03h			**Reserved**				

Table 5-17 Offset from P_BLK Base Address (PCIDV1 D4h-D5h)

7	6	5	4	3	2	1	0
Offset 00h							
Reserved			THT_EN Throttle enable: Enables clock throttling. 0 = Disable 1 = Enable	CLK_VAL Clock throttle duty: Sets STPCLK# throttling duty cycle. 000 = Reserved 001 = 0-12.5% 010 = 12.5-25% 011 = 25-37.5% 100 = 37.5%-50% 101 = 50-62.5% 110 = 62.5-75% 111 = 75%-87.5%			Reserved
Offset 01h-03h			**Reserved**				
Offset 04h-05h			**Force Power Level 2 or 3 Register**				
Bits [15:0] correspond to: [7:0] = 04h, [15:8] = 05h - Bits [7:0] = P_LVL2 Force Power Level 2: Reading this register forces clock control logic to C2 state. Writes are ignored. (SYSCFG 50h[3] - APM Doze Mode) - Bits [15:8] = P_LVL3 Force Power Level 3: Reading this register forces clock control logic to C3 state. Writes are ignored.(SYSCFG 50h[0] - 0V CPU Suspend Mode)							

Table 5-18 Offset from GPE0_BLK Base Address (PCIDV1 D6h-D7h)

7	6	5	4	3	2	1	0
Offset 00h	**GP_STS Register - Byte 0**						
ACPI7 LID_STS Lid open or close event status: 0 = No activity 1 = Activity Write 1 to clear	ACPI6 EC_STS Embedded controller event status: Set if EC# line goes low. 0 = No activity 1 = Input active Write 1 to clear	ACPI5 USB_STS USB# signal status: Set if USB# line goes low. 0 = No activity 1 = Input active Write 1 to clear	ACPI4 RI_STS RI# signal status: Set if RI# goes low (local pin or from IRQ drive-back). 0 = No activity 1 = Input active Write 1 to clear	ACPI3 FRI#_STS FRI# signal status: Set if FRI# goes low (local pin or from IRQ drive-back). 0 = No activity 1 = Input active Write 1 to clear	ACPI2 STSCHG_STS STSCHG# signal status: Set if STSCHG# goes low (provided from a PCM-CIA controller). 0 = No activity 1 = Input active Write 1 to clear	ACPI1 DOCK_STS DOCK# signal status: Set if DOCK# goes low (provided from a docking controller). 0 = No activity 1 = Input active Write 1 to clear	ACPI0 UNDOCK_STS UNDOCK# signal status: Set if UNDOCK# goes low (usually provided from a switch that is either local or on the docking station). 0 = No activity 1 = Input active Write 1 to clear
Offset 01h	**GP_STS Register - Byte 1**						
Reserved			THRM_STS Has bit of the THFREQ value specified in TEMPGR[3:0] toggled? 0 = No 1 = Yes	ACPI11_STS: 0 = No activity 1 = Input active Write 1 to clear	ACPI10_STS: 0 = No activity 1 = Input active Write 1 to clear	ACPI9_STS: 0 = No activity 1 = Input active Write 1 to clear	ACPI8_STS: 0 = No activity 1 = Input active Write 1 to clear
Offset 02h	**GP_EN Register - Byte 0**						
ACPI7 LID_EN Allow SCI on LID_STS event: 0 = No 1 = Yes	ACPI6 EC#_EN Allow SCI from EC#_STS event: 0 = No 1 = Yes	ACPI5 USB#_EN Allow SCI from USB#_STS event: 0 = No 1 = Yes	ACPI4 RI#_EN Allow SCI from RI#_STS event: 0 = No 1 = Yes	ACPI3 FRI#_EN Allow SCI from FRI#_STS event: 0 = No 1 = Yes	ACPI2 STSCHG#_EN Allow SCI from STSCHG#_ STS event: 0 = No 1 = Yes	ACPI1 DOCK#_EN Allow SCI from DOCK#_STS event: 0 = No 1 = Yes	ACPI0 UNDOCK#_EN Allow SCI from UNDOCK#_ STS event: 0 = No 1 = Yes
Offset 03h	**GP_EN Register - Byte 1**						
BIOS_RLS: Does BIOS want to make a request to ACPI? 0 = No effect 1 = Yes Write-only bit; reads always return 0.	Reserved		THRM_EN Allow SCI from THRM_STS event: 0 = No 1 = Yes	ACPI11_EN: Allow SCI from ACPI11_STS event: 0 = No 1 = Yes	ACPI10_EN: Allow SCI from ACPI10_STS event: 0 = No 1 = Yes	ACPI9_EN: Allow SCI from ACPI9_STS event: 0 = No 1 = Yes	ACPI8_EN: Allow SCI from ACPI8_STS event. 0 = No 1 = Yes
Offset 04h-07h	**Reserved**						

5.6 Register Space Summary

This summary includes only System Control and PCI Configuration Register Spaces. For information on ISA-Compatible I/O Registers and ACPI I/O Registers refer to Section 5.5, I/O Register Space. For information on Primary and Secondary IDE I/O Registers, refer to Section 5.4.2, IDE I/O Registers.

Table 5-19 SYSCFG 00h-FFh Register Summary

Loc.	Register Name	Default
00h	Byte Merge/Prefetch & Sony Cache Module Control Register	00h
01h	DRAM Control Register 1	00h
02h	Cache Control Register 1	00h
03h	Cache Control Register 2	00h
04h	Shadow RAM Control Register 1	00h
05h	Shadow RAM Control Register 2	00h
06h	Shadow RAM Control Register 3	00h
07h	Tag Test Register	00h
08h	CPU Cache Control Register	00h
09h	System Memory Function Register	00h
0Ah	DRAM Hole A Address Decode Register	00h
0Bh	DRAM Hole B Address Decode Register	00h
0Ch	DRAM Hole Higher Address	00h
0Dh	Clock Control Register	00h
0Eh	PCI Master Burst Control Register 1	00h
0Fh	PCI Master Burst Control Register 2	00h
10h	Miscellaneous Control Register 1	00h
11h	Miscellaneous Control Register 2	00h
12h	Refresh Control Register	00h
13h	Memory Decode Control Register 1	00h
14h	Memory Decode Control Register 2	00h
15h	PCI Cycle Control Register 1	00h
16h	Dirty/Tag RAM Control Register	A0h
17h	PCI Cycle Control Register 2	00h
18h	Interface Control Register	00h
19h	Memory Decode Control Register 3	00h
1Ah	Memory Shadow Control Register 1	00h
1Bh	Memory Shadow Control Register 2	00h
1Ch	EDO DRAM Control Register	00h
1Dh	Miscellaneous Control Register 3	00h
1Eh	Control Register	00h
1Fh	EDO Timing Control Register	00h
20h	DRAM Burst Control Register	00h
21h	PCI Concurrency Control Register	01h
22h	Inquire Cycle Control Register	00h
23h	Pre-Snoop Control Register	00h
24h	Asymmetric DRAM Configuration Register	00h
25h	GUI Memory Location Register	00h
26h	UMA Control Register 1	00h
27h	Miscellaneous Control Register 4	00h
28h	SDRAM Control Register 1	00h
29h	SDRAM Control Register 2	00h
2Ah	PCI-to-DRAM Control Register 1	00h
2Bh	PCI-to-DRAM Control Register 2	00h
2Ch	CPU-to-DRAM Buffer Control Register	00h
2Dh	Miscellaneous Control Register 5	00h
2Eh	UMA Control Register 2	00h
2Fh	UMA Control Register 3	00h
30h-37h	Reserved	00h
38h	NMI Trap Enable Register 1	00h
39h	NMI Trap Enable Register 2	00h
3Ah	NMI Trap Enable Register 3	00h
3Bh	NMI Trap Enable Register 4	00h
3Ch	NMI Trap Enable Register 5	00h
3Dh-3Fh	Reserved	00h
40h	PMU Control Register 1	00h
41h	DOZE_TIMER Register	00h
42h	If AEh[7] = 0: Clock Source Register 1	00h
	If AEh[7] = 1: Clock Source Register 1A	00h
43h	PMU Control Register 2	00h
44h	LCD_TIMER Register	00h
45h	DSK_TIMER Register	00h
46h	KBD_TIMER Register	00h
47h	If AEh[7] = 0: GNR1_TIMER Register	00h
	If AEh[7] = 1: GNR5_TIMER Register	00h
48h	If AEh[7] = 0: GNR1 Base Address Register	00h
	If AEh[7] = 1: GNR5_Timer Base Address Register	00h
49h	If AEh[7] = 0: GNR1 Control Register	00h
	If AEh[7] = 1: GNR5_Timer Control Register	00h
4Ah	Chip Select 0 Base Address Register	00h
4Bh	Chip Select 0 Control Register	00h
4Ch	Chip Select 1 Base Address Register	00h
4Dh	Chip Select 1 Control Register	00h
4Eh	If AEh[7] = 0: Idle Reload Event Enable Register 1	00h
	If AEh[7] = 1: Idle Reload Event Enable Register 1A	00h

Table 5-19 SYSCFG 00h-FFh Register Summary (cont.)

Loc.	Register Name	Default
4Fh	IDLE_TIMER Register	00h
50h	PMU Control Register 3	00h
51h	Beeper Control Register	00h
52h	Scratchpad Register 1	00h
53h	Scratchpad Register 2	00h
54h	Power Control Latch Register 1	00h
55h	Power Control Latch Register 2	0Fh
56h	Reserved	00h
57h	PMU Control Register 4	08h
58h	PMU Event Register 1	00h
59h	PMU Event Register 2	00h
5Ah	If AEh[7] = 0: PMU Event Register 3	00h
	If AEh[7] = 1: PMU Event Register 3A	00h
5Bh	If AEh[7] = 0: PMU Event Register 4	00h
	If AEh[7] = 1: PMU Event Register 4A	00h
5Ch	PMI SMI Source Register 1 (Write 1 to Clear)	00h
5Dh	If AEh[7] = 0: PMI SMI Source Register 2 (Write 1 to Clear)	00h
	If AEh[7] = 1: PMI SMI Source Register 2A (Write 1 to Clear)	00h
5Eh	Reserved	00h
5Fh	PMU Control Register 5	00h
60h	R_Timer Count Register	00h
61h	Debounce Register	00h
62h	IRQ Doze Register 1	00h
63h	Idle Time-Out Select Register 1	00h
64h	INTRGRP IRQ Select Register 1	00h
65h	Doze Register	00h
66h	PMU Control Register 6	00h
67h	PMU Control Register 7	00h
68h	Clock Source Register 2	00h
69h	R_TIMER Register	00h
6Ah	RSMGRP IRQ Register 1	00h
6Bh	Resume Source Register	00h
6Ch	Scratchpad Register 3	00h
6Dh	Scratchpad Register 4	00h
6Eh	Scratchpad Register 5	00h
6Fh	Scratchpad Register 6	00h
70h	GNR1 Base Address Register 1	00h
71h	GNR1 Control Register 1	FFh
72h	GNR1 Control Register 2	00h
73h	GNR2 Base Address Register 1	00h
74h	GNR2 Control Register 1	FFh
75h	GNR2 Control Register 2	00h

Loc.	Register Name	Default
76h	If AEh[7] = 0: Doze Reload Select Register 1	0Fh
	If AEh[7] = 1: Doze Reload Select Register 1A	03h
77h	Doze Reload Select Register 2	00h
78h	Doze Reload Select Register 3	00h
79h	PMU Control Register 8	00h
7Ah	GNR3 Base Address Register 1	00h
7Bh	GNR3 Control Register 1	FFh
7Ch	GNR3 Control Register 2	00h
7Dh	GNR4 Base Address Register 1	00h
7Eh	GNR4 Control Register 1	FFh
7Fh	GNR4 Control Register 2	00h
80h	ICW1 Shadow Register for INTC1	00h
81h	ICW2 Shadow Register for INTC1	00h
82h	ICW3 Shadow Register for INTC1	00h
83h	ICW4 Shadow Register for INTC1	00h
84h	DMA In-Progress Register (RO)	00h
85h	OCW2 Shadow Register for INTC1	00h
86h	OCW3 Shadow Register for INTC1	00h
87h	Reserved	00h
88h	ICW1 Shadow Register for INTC2	00h
89h	ICW2 Shadow Register for INTC2	00h
8Ah	ICW3 Shadow Register for INTC2	00h
8Bh	ICW4 Shadow Register for INCT2	00h
8Ch	Reserved	00h
8Dh	OCW2 Shadow Register for INTC2	00h
8Eh	OCW3 Shadow Register for INTC2	00h
8Fh	Reserved	00h
90h	Timer Channel 0 Low Byte Register: A[7:0]	00h
91h	Timer Channel 0 High Byte Register: A[15:8]	00h
92h	Timer Channel 1 Low Byte Register: A[7:0]	00h
93h	Timer Channel 1 High Byte Register: A[15:8]	00h
94h	Timer Channel 2 Low Byte Register: A[7:0]	00h
95h	Timer Channel 2 High Byte Register: A[15:8]	00h
96h	Write Counter High/Low Byte Latch (RO)	xxh
97h	Reserved	00h
98h	RTC Index Shadow Register (RO)	xxh
99h	Interrupt Request Register for INCT1 (RO)	xxh
9Ah	Interrupt Request Register for INCT2 (RO)	xxh
9Bh	3F2h + 3F7h Shadow Register	00h
9Ch	372h + 377h Shadow Register	00h
9Dh-9Eh	Reserved	00h
9Fh	Port 064h Shadow Register	00h
A0h	Feature Control Register 1	80h

Table 5-19 SYSCFG 00h-FFh Register Summary (cont.)

Loc.	Register Name	Default
A1h	Feature Control Register 2	00h
A2h	If AEh[7] = 0: IRQ Doze Register 2	00h
	If AEh[7] = 1: IRQ Doze Register 2A	00h
A3h	Idle Time-Out Select Register 2	00h
A4h	INTRGRP IRQ Select Register 2	00h
A5h	Thermal Management Register 1	00h
A6h	Thermal Management Register 2	00h
A7h	Thermal Management Register 3	00h
A8h	Thermal Management Register 4	00h
A9h	Thermal Management Register 5	00h
AAh	Thermal Management Register 6	00h
ABh	Power Control Latch Register 3	00h
ACh	Reserved	00h
ADh	Feature Control Register 3	00h
AEh	GNR_ACCESS Feature Register 1	03h
AFh-B0h	Reserved	00h
B1h	RSMGRP IRQ Register 2	00h
B2h	If AEh[7] = 0: Clock Source Register 3	00h
	If AEh[7] = 1: Clock Source Register 3A	00h
B3h	Chip Select Cycle Type Register	00h
B4h	HDU_TIMER Register	00h
B5h	COM1_TIMER Register	00h
B6h	COM2_TIMER Register	00h
B7h	If AEh[7] = 0: GNR2_TIMER Register	00h
	If AEh[7] = 1: GNR6_TIMER Register	00h
B8h	If AEh[7] = 0: GNR2 Base Address Register	00h
	If AEh[7] = 1: GNR6 Base Address Register	00h
B9h	If AEh[7] = 0: GNR2 Control Register	00h
	If AEh[7] = 1: GNR6 Control Register	00h
BAh	Chip Select 2 Base Address Register	00h\
BBh	Chip Select 2 Control Register	00h
BCh	Chip Select 3 Base Address Register	00h
BDh	Chip Select 3 Control Register	00h
BEh	If AEh[7] = 0: Idle Reload Event Enable Register 2	00h
	If AEh[7] = 1: Idle Reload Event Enable Register 2A	00h
BFh	Chip Select Granularity Register	0Fh
C0h-D4h	Reserved	00h
D5h	X Bus Positive Decode Register	00h
D6h	PMU Control Register 9	00h
D7h	Access Port Address Register 1	00h
D8h	If AEh[7] = 0: PMU Event Register 5	00h
	If AEh[7] = 1: PMU Event Register 5A	00h

Loc.	Register Name	Default
D9h	PMU Event Register 6	00h
DAh	Power Management Event Status Register (RO)	00h
DBh	If AEh[7] = 0: Next Access Event Generation Register 1	00h
	If AEh[7] = 1: Next Access Event Generation Register 1A	00h
DCh	If AEh[7] = 0: PMU SMI Source Register 1 (Write 1 to Clear)	00h
	If AEh[7] = 1: PMU SMI Source Register 1A (Write 1 to Clear)	00h
DDh	PMU SMI Source Register 2 (Write 1 to Clear)	00h
DEh	If AEh[7] = 0: Current Access Event Generation Register 1	00h
	If AEh[7] = 1: Current Access Event Generation Register 1A	00h
DFh	If AEh[7] = 0: Activity Tracking Register 1	00h
	If AEh[7] = 1: Activity Tracking Register 1A	00h
E0h	If AEh[7] = 0: Activity Tracking Register 2	00h
	If AEh[7] = 1: Activity Tracking Register 2A	00h
E1h	If AEh[7] = 0: GNR3 Base Address Register	00h
	If AEh[7] = 1: GNR7 Base Address Register	00h
E2h	If AEh[7] = 0: GNR3 Control Register	00h
	If AEh[7] = 1: GNR7 Control Register	00h
E3h	If AEh[7] = 0: GNR4 Base Address Register	00h
	If AEh[7] = 1: GNR8 Base Address Register	00h
E4h	If AEh[7] = 0: GNR4 Control Register	00h
	If AEh[7] = 1: GNR8 Control Register	00h
E5h	GNR_ACCESS Feature Register 2	03h
E6h	If AEh[7] = 0: Clock Source Register 4	70h
	If AEh[7] = 1: Clock Source Register 4A	70h
E7h	If AEh[7] = 0: GNR3_TIMER Register	00h
	If AEh[7] = 1: GNR7_TIMER Register	00h
E8h	If AEh[7] = 0: GNR4_TIMER Register	00h
	If AEh[7] = 1: GNR8_TIMER Register	00h
E9h	If AEh[7] = 0: PMU Event Register 7	00h
	If AEh[7] = 1: PMU Event Register 7A	00h
EAh	If AEh[7] = 0: PMU SMI Source Register 3 (Write 1 to Clear)	00h
	If AEh[7] = 1: PMU SMI Source Register 3A (Write 1 to Clear)	00h
EBh	Access Port Address Register 2	00h
ECh	Write Trap Register 1 (RO)	00h
EDh	Write Trap Register 2 (RO)	00h
EEh	Power Control Latch Register 4	0Fh
EFh	Hot Docking Control Register 1	00h
F0h	Hot Docking Control Register 2	00h
F1h	Low Order Start Address for ROM Window	00h

Table 5-19 SYSCFG 00h-FFh Register Summary (cont.)

Loc.	Register Name	Default
F2h	High Order Start Address for ROM Window	00h
F3h	Thermal Management Register 7	00h
F4h	Thermal Management Register 8	00h
F5h	PMU Event Register 8	00h
F6h	DMA Doze Reload Register 1	00h
F7h	DMA Doze Reload Register 2	00h
F8h	Compact ISA Control Register 1	00h
F9h	Compact ISA Control Register 2	00h
FAh	Compact ISA Control Register 3	00h
FBh	DMA Idle Reload Register	00h
FCh	IDE Power Management Assignment Register 1	33h
FDh	IDE Power Management Assignment Register 2	33h
FEh	GPCS# Global Control Register	00h
FFh	Reserved	00h

Table 5-20 PCIDV0 00h-FFh Register Summary

Loc.	Register Name	Default
00h	Vendor Identification Register (RO) - Byte 0	45h
01h	Vendor Identification Register (RO) - Byte 1	10h
02h	Device Identification Register (RO) - Byte 0	01h
03h	Device Identification Register (RO) - Byte 1	C7h
04h	Command Register - Byte 0	07h
05h	Command Register - Byte 1	00h
06h	Status Register - Byte 0	80h
07h	Status Register - Byte 1	00h
08h	Revision Identification Register (RO)	10h
09h	Class Code Register (RO) - Byte 0	00h
0Ah	Class Code Register (RO) - Byte 1	00h
0Bh	Class Code Register (RO) - Byte 2	06h
0Ch	Reserved	00h
0Dh	Master Latency Timer Register (RO)	00h
0Eh	Header Type Register (RO)	00h
0Fh	Built-In Self-Test (BIST) Register (RO)	00h
10h-2Bh	Reserved	00h
2Ch-2Dh	Subsystem Vendor ID	00h
2Eh-2Fh	Subsystem ID	00h
30h-3Fh	Reserved	00h
40h	Memory Control Register - Byte 0	00h
41h	Memory Control Register - Byte 1	00h
42h	Reserved	00h
43h	Internal Project Revision - Reserved	00h
44h	Data Path Register 1	00h
45h	Data Path Control Register 2	00h
46h	Data Path Control Register 3	00h
47h	Data Path Control Register 4	00h
48h	Data Path Control Register 5	00h
49h-4Bh	Reserved	00h
4Ch	MCACHE Control Register	00h
4Dh	Delay Adjustment Register	00h
4Eh	SDRAM Control Register	00h
4Fh-FFh	Reserved	00h

Table 5-21 PCIDV1 00h-FFh Register Summary

Loc.	Register Name	Default
00h	Vendor Identification Register (RO) - Byte 0	45h
01h	Vendor Identification Register (RO) - Byte 1	10h
02h	Device Identification Register (RO) - Byte 0	00h
03h	Device Identification Register (RO) - Byte 1	C7h
04h	Command Register - Byte 0	07h
05h	Command Register - Byte 1	00h
06h	Status Register - Byte 0	80h
07h	Status Register - Byte 1	02h
08h	Revision Identification Register (RO)	10h
09h	Class Code Register (RO) - Byte 0	00h
0Ah	Class Code Register (RO) - Byte 1	00h
0Bh	Class Code Register (RO) - Byte 2	06h
0Ch	Reserved	00h
0Dh	Master Latency Timer Register (RO)	00h
0Eh	Header Type Register (RO)	00h
0Fh	Built-In Self-Test (BIST) Register (RO)	00h
10h-2Bh	Reserved	00h
2Ch-2Dh	Subsystem Vendor ID	00h
2Eh-2Fh	Subsystem ID	00h
30h-40h	Reserved	00h
41h	Keyboard Controller Select Register	00h
42h	Reserved	00h
43h	Feature Control Register	00h
44h-45h	Reserved	00h
46h	PCI Control Register B - Byte 0	06h
47h	PCI Control Register B - Byte 1	00h
48h	Strap Option Readback Register - Byte 0	00h
49h	Strap Option Readback Register - Byte 1	00h
4Ah	ROM Chip Select Register 1	00h
4Bh	ROM Chip Select Register 2	00h
4Ch-4Dh	Reserved	00h
4Eh	Miscellaneous Control Register 1	00h
4Fh	Miscellaneous Control Register 2	20h
50h-51h	Reserved	00h
52h	Miscellaneous Controller Register 3	00h
53h	Miscellaneous Controller Register 4	00h
54h	IRQ Driveback Address Register - Byte 0: Address Bits [7:0]	00h
55h	IRQ Driveback Address Register - Byte 1: Address Bits [15:8]	00h
56h	IRQ Driveback Address Register - Byte 2: Address Bits [23:16]	00h
57h	IRQ Driveback Address Register - Byte 3: Address Bits [31:24]	00h
58h	DRQ Remap Base Address Register - Byte 0: Address Bits [7:0]	00h
59h	DRQ Remap Base Address Register - Byte 1: Address Bits [15:8]	00h
5Ah	DRQ Remap Base Address Register - Byte 2: Address Bits [23:16]	00h
5Bh	DRQ Remap Base Address Register - Byte 3: Address Bits [31:24]	00h
5Ch	DMA Channel Selector Register	00h
5Dh	Reserved	00h
5Eh	IRQ Scheme Management Register	00h
5Fh	SYSCFG Base Select Register	00h
60h	IRQ Driveback Data Register - Byte 0: Data Bits [7:0]	00h
61h	IRQ Driveback Data Register - Byte 1: Data Bits [15:8]	00h
62h	IRQ Driveback Data Register - Byte 2: Data Bits [23:16]	00h
63h	IRQ Driveback Data Register - Byte 3: Data Bits [31:24]	00h
64h	PCI Master Control Register 1	10h
65h	PCI Master Control Register 2	01h
66h	Reserved	00h
67h	Miscellaneous Control Register 5	00h
68h	PCICLK Control Register 1	FFh
69h	PCICLK Control Register 2	00h
6Ah	PCICLK Skew Adjust Register for PCICLK 0, 1, 2	00h
6Bh	PCICLK Skew Adjust Register for PCICLK 3, 4, 5	00h
6Ch-6Fh	Reserved	00h
70h	Leakage Control Register - Byte 0	00h
71h	Leakage Control Register - Byte 1	00h
72h	Leakage Control Register - Byte 2	00h
73h	Leakage Control Register - Byte 3	00h
74h	Leakage Control Register - Byte 4	00h
75h	Leakage Control Register - Byte 5	00h
76h	Hot Docking Leakage Control Register	00h
77h-7Fh	Reserved	00h

Table 5-21 PCIDV1 00h-FFh Register Summary (cont.)

Loc.	Register Name	Default
80h	PIO0 Pin (CDOE#) Function Register	00h
81h	PIO1 Pin (TAGWE#) Function Register	00h
82h	PIO2 Pin (ADSC#) Function Register	00h
83h	PIO3 Pin (ADV#) Function Register	00h
84h	PIO4 Pin (RAS2#) Function Register	00h
85h	PIO5 Pin (RAS1#) Function Register	00h
86h	PIO6 Pin (CLKRUN#) Function Register	00h
87h	PIO7 Pin (REQ1#) Function Register	00h
88h	PIO8 Pin (REQ2#) Function Register	00h
89h	PIO9 Pin (DDRQ0) Function Register	00h
8Ah	PIO10 Pin (IRQ1) Function Register	00h
8Bh	PIO11 Pin (IRQ8#) Function Register	00h
8Ch	PIO12 Pin (IRQ12) Function Register	00h
8Dh	PIO13 Pin (IRQ14) Function Register	00h
8Eh	PIO14 Pin (SEL#/ATB#) Function Register	00h
8Fh	PIO15 Pin (RSTDRV) Function Register	00h
90h	PIO16 Pin (SA16) Function Register	00h
91h	PIO17 Pin (SA17) Function Register	00h
92h	PIO18 Pin (IO16#) Function Register	00h
93h	PIO19 Pin (M16#) Function Register	00h
94h	PIO20 Pin (SBHE#) Function Register	00h
95h	PIO21 Pin (SMRD#) Function Register	00h
96h	PIO22 Pin (SMWR#) Function Register	00h
97h	PIO23 Pin (ROMCS#) Function Register	00h
98h	PIO24 Pin (KBDCS#) Function Register	00h
99h	PIO25 Pin (DRQA) Function Register	00h
9Ah	PIO26 Pin (DRQB) Function Register	00h
9Bh	PIO27 Pin (DRQC) Function Register	00h
9Ch	PIO28 Pin (DRQD) Function Register	00h
9Dh	PIO29 Pin (DRQE) Function Register	00h
9Eh	PIO30 Pin (DRQF) Function Register	00h
9Fh	PIO31 Pin (DRQG) Function Register	00h
A0h	Logic Matrix Register 1	00h
A1h	Logic Matrix Register 2	00h
A2h	Logic Matrix Register 3	00h
A3h	Logic Matrix Register 4	00h
A4h	Logic Matrix Register 5	00h
A5h	Logic Matrix Register 6	00h
A6h	Logic Matrix Register 7	00h
A7h	Logic Matrix Register 8	00h
A8h	PIO Pin Current State Register 1	00h
A9h	PIO Pin Current State Register 2	00h
AAh	PIO Pin Current State Register 3	00h
ABh	PIO Pin Current State Register 4	00h

Loc.	Register Name	Default
ACh-ADh	Reserved	00h
AEh	DBE# Select Register 1	01h
AFh	DBE# Select Register 2	00h
B0h	IRQA Interrupt Selection Register	03h
B1h	IRQB Interrupt Selection Register	04h
B2h	IRQC Interrupt Selection Register	05h
B3h	IRQD Interrupt Selection Register	06h
B4h	IRQE Interrupt Selection Register	07h
B5h	IRQF Interrupt Selection Register	09h
B6h	IRQG Interrupt Selection Register	0Ah
B7h	IRQH Interrupt Selection Register	0Bh
B8h	PCI Interrupt Selection Register 1	00h
B9h	PCI Interrupt Selection Register 2	00h
BAh	Serial IRQ Control Register 1	00h
BBh	Serial IRQ Control Register 2	00h
BCh-BFh	Reserved	00h
C0h	DMA Channels A and B Selection Register	10h
C1h	DMA Channels C and D Selection Register	32h
C2h	DMA Channel E Selection Register	50h
C3h	DMA Channels F and G Selection Register	76h
C4h-CFh	Reserved	00h
Note: The registers located from PCIDV1 D0h through EEh pertain only to FS ACPI Version. Otherwise they are reserved.		
D0h	FS ACPI: PM1_BLK Base Address Register - Byte 0: Address Bits [7:0]	00h
D1h	FS ACPI: PM1_BLK Base Address Register - Byte 1: Address Bits [15:8]	00h
D2h	FS ACPI: PM2_BLK Base Address Register - Byte 0: Address Bits [7:0]	00h
D3h	FS ACPI: PM2_BLK Base Address Register - Byte 1: Address Bits [15:8]	00h
D4h	FS ACPI: P_BLK Base Address Register - Byte 0: Address Bits [7:0]	00h
D5h	FS ACPI: P_BLK Base Address Register - Byte 1: Address Bits [15:8]	00h
D6h	FS ACPI: GPE0_BLK Base Address Register - Byte 0: Address Bits [7:0]	00h
D7h	FS ACPI: GPE0_BLK Base Address Register - Byte 0: Address Bits [15:8]	00h
D8h	FS ACPI: ACPI Source Control Register - Byte 0	00h
D9h	FS ACPI: ACPI Source Control Register - Byte 1	00h
DAh	FS ACPI: ACPI Source Status Register - Byte 0	00h

Table 5-21 PCIDV1 00h-FFh Register Summary (cont.)

Loc.	Register Name	Default
DBh	FS ACPI: ACPI Source Status Register - Byte 1	00h
DCh	FS ACPI: ACPI Event Resume Control Register - Byte 0	00h
DDh	FS ACPI: ACPI Event Resume Control Register - Byte 1	00h
DEh-DFh	Reserved	00h
E0h	FS ACPI: SLP_TYP Control Register - Byte 0	00h
E1h	FS ACPI: SLP_TYP Control Register - Byte 1	00h
E2h	FS ACPI: SLP_TYP Control Register - Byte 2	00h
E3h	FS ACPI: SLP_TYP Control Register - Byte 3	00h
E4h	FS ACPI: SLP_TYP Control Register - Byte 4	00h
E5h	FS ACPI: SLP_TYP Control Register - Byte 5	00h
E6h	FS ACPI: SLP_TYP Control Register - Byte 6	00h
E7h	FS ACPI: SLP_TYP Control Register - Byte 7	00h
E8h	FS ACPI: Power Control Latch Set Register	00h
E9h	Reserved	00h
EAh	FS ACPI: Power Control Readback Register - Byte 0	FFh
EBh	FS ACPI: Power Control Readback Register - Byte 1	FFh
ECh	FS ACPI: Power Control Readback Register - Byte 2	F0h
EDh	FS ACPI: Power Control Readback Register - Byte 3	F0h
EEh	FS ACPI: ACPI Thermal Control Register	00h
EFh-FDh	Reserved	00h
FEh	Stop Grant Cycle Generation Register (WO)	00h
FFh	Parity Error Cycle Generation Register	00h

Table 5-22 PCIIDE 00h-47h Register Summary

Loc	Register Name	Default
00h	Vendor ID Register (RO) - Byte 0	45h
01h	Vendor ID Register (RO) - Byte 1	10h
02h	Device ID Register (RO) - Byte 0 (SYSCFG ADh[2] controls the value returned by this register.)	68h
03h	Device ID Register (RO) - Byte 1 (SYSCFG ADh[2] controls the value returned by this register.)	D5h
04h	Command Register - Byte 0	45h
05h	Command Register (RO) - Byte 1	00h
06h	Status Register (RO) - Byte 0	80h
07h	Status Register - Byte 1	02h
08h	Revision ID Register (RO)	00h
09h	Class Code Register - Byte 0	80h
0Ah	Class Code Register (RO) - Byte 1	01h
0Bh	Class Code Register (RO) - Byte 2	01h
0Ch-0Dh	Reserved	00h
0Eh	Header Type Register (RO)	00h
0Fh	Built-In Self-Test Register (RO)	00h
10h-13h	Primary IDE Command Block Base Address Register	1F1h with PCIIDE 09h[2] = 1 and 40h[2] = 1
14h-17h	Primary IDE Control Block Base Address Register	3F5h with PCIIDE 09h[2] = 1 and 40h[2] = 1
18h-1Bh	Secondary IDE Command Block Base Address Register	171h with PCIIDE 09h[2] =1, 40h[2] = 1, and 40h[3] = 0
1Ch-1Fh	Secondary IDE Control Block Base Address Register	375h with PCIIDE 09h[2] = 1, 40h[2] = 1, and 40h[3] = 0
20h-23h	Bus Master IDE Base Address Register	00000001h
24h-2Bh	Reserved	00h
2Ch-2Dh	Subsystem Vendor ID (write one time only)	00h
2Eh-2Fh	Subsystem ID (write one time only)	00h
30h-3Ah	Reserved	00h
3Ch	Interrupt Line Register	00h
3Dh	Interrupt Pin Register (RO)	FFh
3Eh-3Fh	Reserved	00h
40h	IDE Initialization Control Register	00h
41h	Reserved	00h
42h	IDE Enhanced Feature Register	00h
	FS ACPI: IDE Enhanced Feature Register	00h
43h	IDE Enhanced Mode Register	00h
44h	Emulated Bus Master Register	00h
	FS ACPI: Ultra DMA Configuration Register	00h
45h	IDE Interrupt Selection Register	00h
	FS ACPI: Ultra DMA Configuration Register	00h
46h	FS ACPI: Emulated IDE Configuration Register	00h
47h	FS ACPI: IDE Interrupt Selection Register	FAh

6.0 Electrical Ratings

Stresses above those listed in the following tables may cause permanent damage to the device. These are stress ratings only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification are not implied.

6.1 Absolute Maximum Ratings

Symbol	Parameter	5.0 Volt		3.3 Volt		Unit
		Min	Max	Min	Max	
VCC	5.0V Supply Voltage		+6.5			V
VDD	3.3V Supply Voltage				+4.0	
VI	Input Voltage	–0.5	VCC + 0.5	–0.5	VDD + 0.5	V
VO	Output Voltage	–0.5	VCC + 0.5	–0.5	VDD + 0.5	V
TOP	Operating Temperature	0	+85	0	+85	°C
TSTG	Storage Temperature	–40	+125	–40	+125	°C

6.2 DC Characteristics:

VCC_PCI, VCC_DRAM, VCC_CPU = 3.3V±5%, VCC_ISA = 5.0V±5%, TA = 0°C to +85°C

Symbol	Parameter	Min	Max	Unit	Condition
VIL	Input low Voltage	–0.5	+0.8	V	
VIH	Input high Voltage	+2.0	VCC + 0.5	V	
VOL	Output low Voltage		+0.4	V	IOL = 4.0mA
VOH	Output high Voltage	+2.4		V	IOH = -1.6mA
IIL	Input Leakage Current		+10.0	µA	VIN = VCC
IOZ	Tristate Leakage Current		+10.0	µA	
CIN	Input Capacitance		+10.0	pF	
COUT	Output Capacitance		+10.0	pF	
ICC	Power Consumption		1	W	At 66MHz, full operation
θJC	Junction to Case Thermal Resistance	<5		C°/W	
θAC	Case to Ambient Thermal Resistance	22		C°/W	

Note: Average power dissipation for a system running at 60/90MHz: less than 1.6W (estimated).

6.3 Data Buffer Controller Module AC Characteristics (66MHz - Preliminary)

Symbol	Parameter	Min	Max	Unit	Condition
t101	HD[63:0] to MD[63:32] bus valid	2	22	ns	
t102	HD[31:0] to MD[31:0]# bus valid	2	22	ns	
t107	MD[31:0] setup to DLE# low	5		ns	
t108	MD[63:32] setup to DLE# low	5		ns	
t110	HD[31:0] setup to DLE# low	5		ns	
t111	HD[63:32] setup to DLE# low	5		ns	
t115	MD[31:0] hold from DLE# high	5		ns	
t116	MD[63:32] hold from DLE# high	5		ns	
t118	HD[31:0] hold from DLE# high	8		ns	
t119	HD[63:32] hold from DLE# high	8		ns	

Note: DLE# is an internal signal (timing TBD).

6.4 CPU Interface Module AC Characteristics (66MHz - Preliminary)

Symbol	Parameter	Min	Max	Unit	Condition
t400	BOFF# valid delay from CPUCLKIN	5	15		
t401	HITM# setup time to CPUCLKIN	2			
t402	HITM# hold time to CPUCLKIN	1			
t406	INV valid delay from CPUCLKIN on (W/R#)/INV signal	5	15		
t407	WB/WT# valid delay from CPUCLKIN on EADS#/(WB/ WT#) signal	5	15		
t201	CPUCLKIN to BRDY# active delay	5	15	ns	
t202	CPUCLKIN to BRDY# inactive delay	5	15	ns	
t205	CDOE# falling edge valid delay from CPUCLKIN rising	5	15	ns	
t207	ADS# setup to CPUCLKIN high	2		ns	
t208	ADS# hold time from CPUCLKIN high	1		ns	
t209	M/IO#, D/C#, W/R#, CACHE# setup to CPUCLKIN high	1		ns	Sampled one CPUCLKIN after ADS#
t408	M/IO#, D/C#, W/R#, CACHE# hold to CPUCLKIN high	1			Sampled one CPUCLKIN after ADS#
t212	CPUCLKIN to TAGWE# active delay	5	14	ns	
t213	CPUCLKIN to TAGWE# inactive delay	5	14	ns	
t214	CACS# falling edge valid delay from CPUCLKIN high	5	15	ns	
t215	BWE#, GWE# falling edge valid delay from CPUCLKIN high	5	15	ns	
t216	CPUCLKIN to NA# active delay	5	15	ns	

6.4 CPU Interface Module AC Characteristics (66MHz - Preliminary) (cont.)

Symbol	Parameter	Min	Max	Unit	Condition
t217	CPUCLKIN to NA# inactive delay	5	15	ns	
t218	TAG[7:0] data read to BRDY# low		5	ns	
t219	CPUCLKIN to ADSC# active delay	5	15	ns	
t220	CPUCLKIN to ADV# active delay	5	15	ns	
t223	HA[31:3] valid delay from PCICLK high	2	18	ns	
t224	HA[31:3] Float delay from PCICLK high	2	18	ns	
t225	AHOLD valid delay from CPUCLKIN high	5	15	ns	
t226	EADS# valid delay from CPUCLKIN high	5	15	ns	
t227	RESET rising edge valid from CPUCLKIN high	5	15	ns	
t228	RESET falling edge valid delay from CPUCLKIN high	5	15	ns	
t229	KEN# valid delay from CPUCLKIN high	5	15	ns	
t409	AHOLD Setup time to PCICLK	5			
t410	AHOLD Hold time to PCICLK	3			
t411	HLDA setup time to PCICLK	5			
t412	HLDA hold time to PCICLK	3			
t413	NMI valid delay from PCICLK	2	15		
t414	RESET setup time to PCICLK	5			
t415	RESET hold time to PCICLK	3			
t313	INIT valid delay from PCICLK rising	2	15	ns	
t315	SMI# valid delay from PCICLK rising	2	15	ns	

Note: BE[7:0]#, LOCK#, SMIACT#, and A[31:0] are not sampled with respect to CPUCLK. These inputs directly feed combinatorial inputs.

6.5 DRAM Controller Module AC Characteristics (66MHz - Preliminary)

Symbol	Parameter	Min	Max	Unit	Condition
t230	RAS[3:0]# valid delay from CPUCLK high/PCICLK high	2	15	ns	
t231	CAS[7:0]# valid delay from CPUCLK high/PCICLK high	2	15	ns	
t232	MA[11:0] valid delay from CPUCLK high/PCICLK high	2	15	ns	
t233	DWE# valid delay from CPUCLK high/PCICLK high	2	15	ns	
t234	MA[11:0] propagation delay from HA[28:3]	2	22	ns	

6.6 PCI Controller Module AC Characteristics (66MHz - Preliminary)

Symbol	Parameter	Min	Max	Unit	Condition
t235	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, DEVSEL# valid delay from PCICLK rising	2	11	ns	
t236	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, PLOCK#, DEVSEL# active to float delay from PCICLK rising	2	15	ns	
t237	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, DEVSEL# setup time to PCICLK rising	7		ns	
t238	C/BE[3:0]#, FRAME#, TRDY#, IRDY#, STOP#, DEVSEL# hold time from PCICLK rising	0		ns	
t239	AD[31:0] valid delay from PCICLK high	2	11	ns	
t240	AD[31:0] setup time to PCICLK high	7		ns	
t241	AD[31:0] hold time from PCICLK high	0		ns	
t301	FRAME#, TRDY#, IRDY#, STOP#, DEVSEL#, PAR, SERR#, PERR# valid delay from PCICLK rising	2	11	ns	
t302	GNT[2:0]# valid delay from PCICLK rising	2	12	ns	
t303	PCIRQ[3:0]# valid delay from PCICLK rising	2	16	ns	
t304	MD[63:32] valid delay from PCICLK rising	2	20	ns	
t305	FRAME#, TRDY#, IRDY#, STOP#, DEVSEL#, PAR, SERR#, PERR# float delay from PCICLK rising	2	20	ns	
t306	C/BE[3:0]#, AD[31:0], FRAME#, IRDY#, TRDY#, STOP#, DEVSEL#, PLOCK#, PAR, SERR#, PERR# setup time to PCICLK rising	7		ns	
t307	C/BE[3:0]#, AD[31:0], FRAME#, IRDY#, TRDY#, STOP#, DEVSEL#, PLOCK#, PAR, SERR#, PERR# hold time from PCICLK rising	0		ns	
t308	PREQ[2:0]# setup time to PCICLK rising	12		ns	
t309	PREQ[2:0]# hold time from PCICLK rising	0		ns	
t310	PCIRQ[3:0]# setup time to PCICLK rising	5		ns	
t311	PCIRQ[3:0]# hold time from PCICLK rising	3		ns	

6.7 ISA Controller Module AC Characteristics (66MHz - Preliminary)

Symbol	Parameter	Min	Max	Unit	Condition
t314	ATCLK rising edge delay from PCICLK rising edge	5	20	ns	
t416	A20M# valid delay from ATCLK	2	15	ns	
t316	IOR#, IOW# high valid delay from ATCLK rising		15	ns	
t317	MRD#, MWR#, SMRD#, SMWR# valid delay from ATCLK rising		15	ns	
t318	BALE low valid delay from ATCLK rising		15	ns	
t320	STPCLK# valid delay from ATCLK rising		15	ns	
t321	RTCAS, RTCRD#, RTCWR#, ROMCS#/KBDCS# valid delay from ATCLK rising		15	ns	
t322	BALE high valid delay from ATCLK falling		15	ns	
t323	IOR#, IOW#, MRD#, MWR#, SMRD#, SMWR# low valid delay from ATCLK falling		15	ns	
t324	PPWR0# valid delay from ATCLK falling		15	ns	
t325	NOWS# setup time to ATCLK falling	0		ns	
t326	NOWS# hold time from ATCLK falling	5		ns	
t327	IOCHRDY setup time to ATCLK falling	5		ns	
t328	IOCHRDY hold time from ATCLK falling	5		ns	

Note: FERR# from the CPU is not sampled with respect to any clock. The input directly feeds combinatorial circuits.

6.8 AC Timing Diagrams

Figure 6-1 Setup Timing Waveform

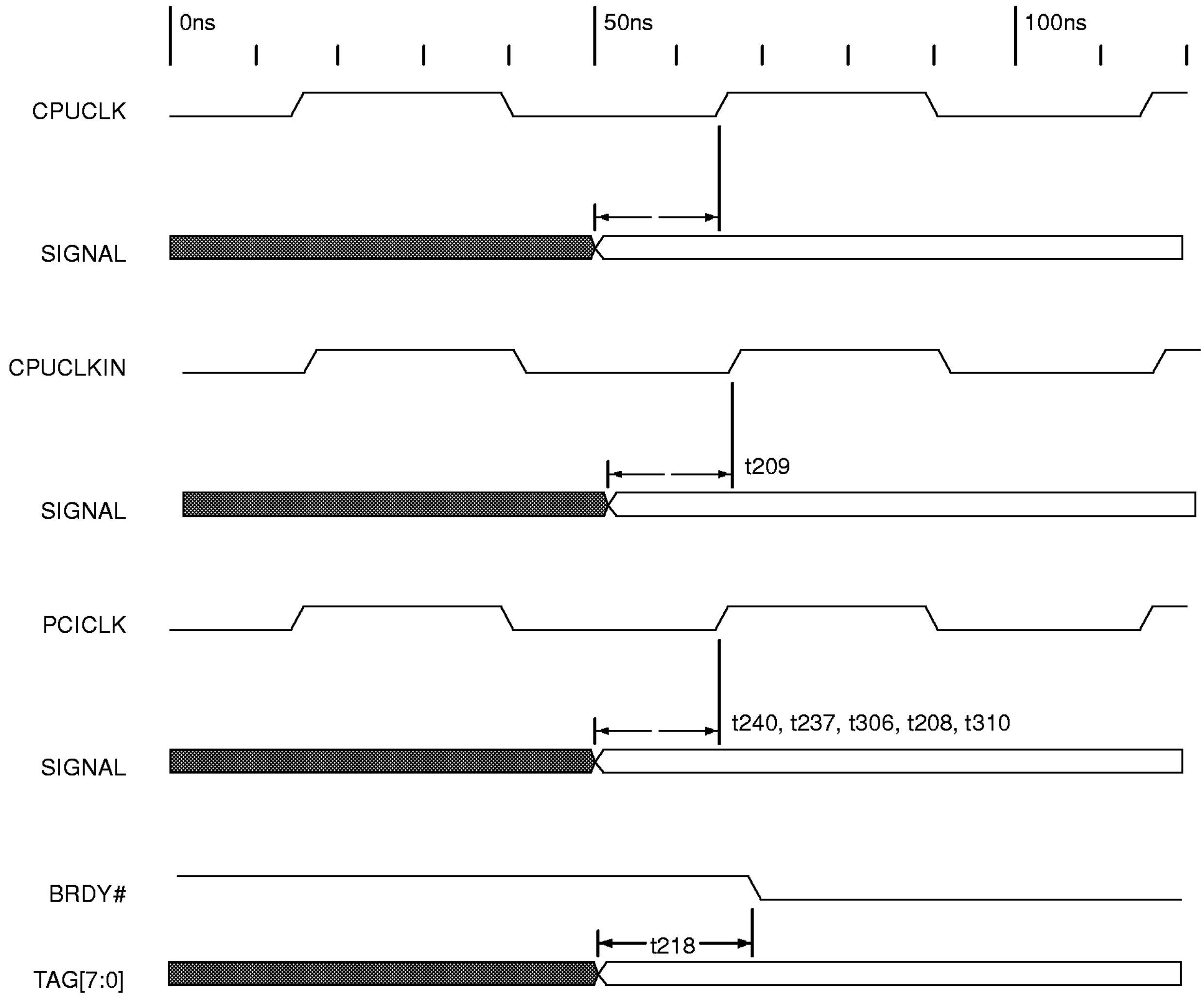


Figure 6-2 Hold Timing Waveform

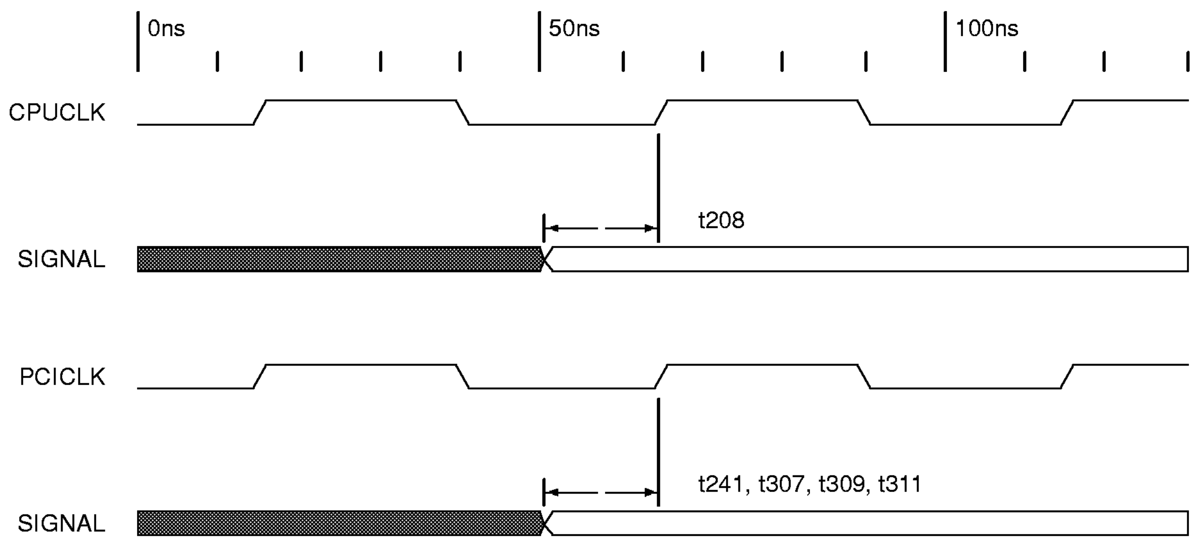


Figure 6-3 Output Delay Timing Waveform

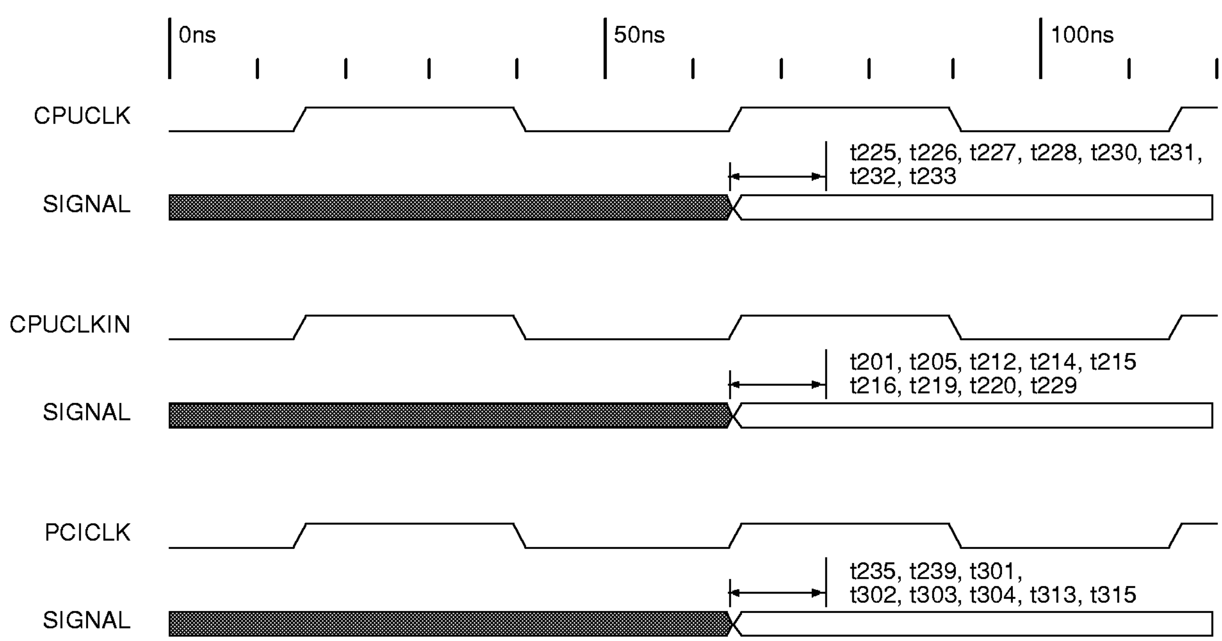
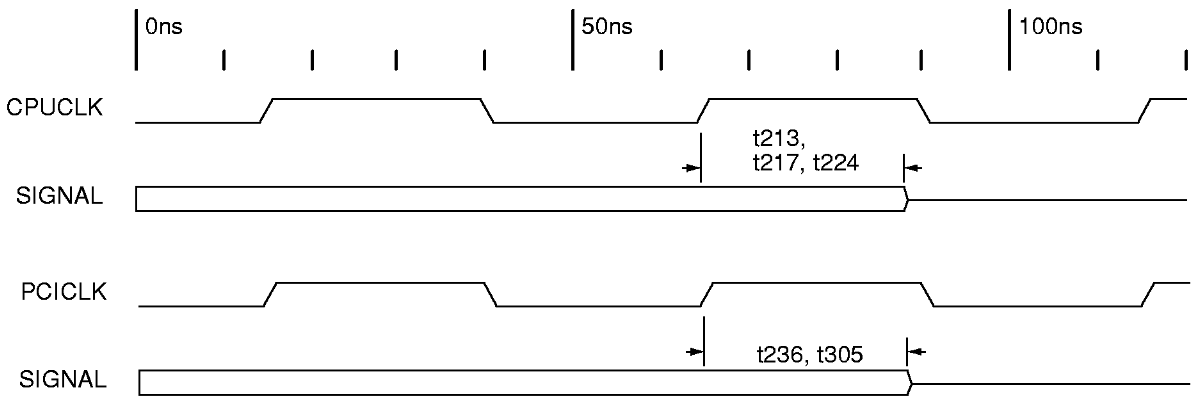


Figure 6-4 Float Delay Timing Waveform



7.0 Test Mode Information

FireStar can be forced into two types of test modes for board-level testing automatic test equipment (ATE).

- Test Mode 0: All output and bidirectional pins are tristated.
- Test Mode 1 (NAND tree test): All bidirectionals are tristated and the end of the input and bidirectional NAND chain is present on pin K22.

Pins AB5, B7, and A7 are used to select between normal and test mode operation. To enable normal operations, pin AB5 must be low. When AB5 is high, FireStar will enter the test mode. Pins B7 and A7 then control which test mode FireStar will enter. Table 7-1 summarizes the mode selection process.

The NAND tree mode is used to test input and bidirectional pins which will be part of the NAND tree chain. The NAND tree chain starts at pin G22 (MD0) and the output of the chain is pin K22 (DACK0#/DACKA#). Table 7-2 gives the pins of the NAND tree chain.

Table 7-1 FireStar Mode Selection

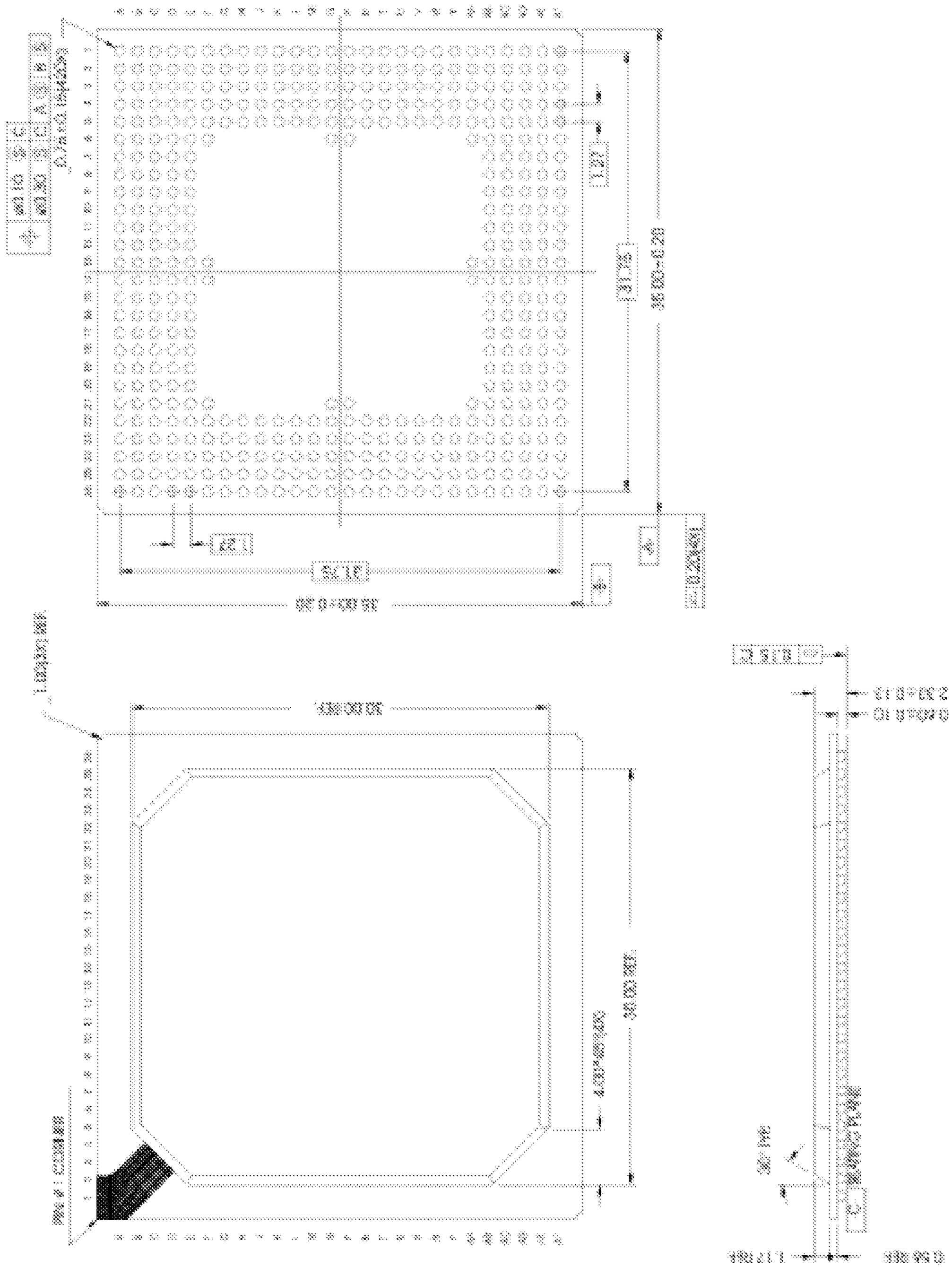
Mode	Pin AB5	Pin B7	Pin A7
Normal Operation (Default)	0	X	X
Test Mode 0 (Tristate Output and Bidirectional Pins)	1	0	0
Test Mode 1 (NAND Tree Test)	1	0	1
Reserved (For Factory Test)	1	1	0
Reserved (For Factory Test)	1	1	1

Table 7-2 NAND Tree Test Mode Pins

Pin #	Remark
G22	MD0 **(NAND Input Start)**
G23	MD1
G24	MD2
G25	MD3
G26	MD4
F22	MD5
F23	MD6
F24	MD7
F25	MD8
F26	MD9
E23	MD10
E24	MD11
E25	MD12
E26	MD13
D24	MD14
D25	MD15
D26	MD16
C25	MD17
C26	MD18
B26	MD19
A26	MD20
B25	MD21
A25	MD22
C24	MD23
B24	MD24
A24	MD25
D23	MD26
C23	MD27
B23	MD28
A23	MD29
D22	MD30
C22	MD31
B22	MD32
A22	MD33
E21	MD34
D21	MD35
C21	MD36
B21	MD37
A21	MD38
D20	MD39
C20	MD40
B20	MD41
A20	MD42
E19	MD43
D19	MD44
C19	MD45
B19	MD46
A19	MD47
E18	MD48
D18	MD49
C18	MD50
B18	MD51
A18	MD52
D17	MD53
C17	MD54
B17	MD55
A17	MD56
E16	MD57
D16	MD58
C16	MD59
B16	MD60
A16	MD61
E15	MD62
D15	MD63
E9	TAG0
D9	TAG1
C9	TAG2
B9	TAG3
A9	TAG4
D8	TAG5
C8	TAG6
B8	TAG7
C7	OSC32
B7	RSVD
A7	SDCKE
E5	OSC_14MHZ
E6	HD63
D6	HD62
C6	HD61
B6	HD60
A6	HD59
D5	HD58
C5	HD57
B5	HD56
A5	HD55
C4	HD54
B4	HD53
A4	HD52
B3	HD51
A3	HD50
A2	HD49
A1	HD48
B2	HD47
B1	HD46
C3	HD45
C2	HD44
C1	HD43
D4	HD42
D3	HD41
D2	HD40
D1	HD39
E4	HD38
E3	HD37
E2	HD36
E1	HD35
F5	HD34
F4	HD33
F3	HD32
F2	HD31
F1	HD30
G4	HD29
G3	HD28
G2	HD27
G1	HD26
H5	HD25
H4	HD24
H3	HD23
H2	HD22
H1	HD21
J5	HD20
J4	HD19
J3	HD18
J2	HD17
J1	HD16
K4	HD15
K3	HD14
K2	HD13
K1	HD12
L5	HD11
L4	HD10
L3	HD9
L2	HD8
L1	HD7
M4	HD6
M3	HD5
M2	HD4
M1	HD3
N4	HD2
N3	HD1
N2	HD0
M5	CPUCLKIN
P3	CACS#
R5	BOFF#
R4	HITM#
R3	A20M#
T3	D/C#
T2	CACHE#
T1	FERR#
U2	LOCK#
U1	SMIACT#
V5	ADS#
V4	BE7#
V3	BE6#
V2	BE5#
V1	BE4#
W4	BE3#
W3	BE2#
W2	BE1#
W1	BE0#
Y5	M/IO#
Y4	HA3
Y3	HA4
Y2	HA5
Y1	HA6
AA5	W/R#
AA4	HA7
AA3	HA8
AA2	HA9
AA1	HA10
AB4	HA11
AB3	HA12
AB2	HA13
AB1	HA14
AC3	HA15
AC2	HA16
AC1	HA17
AD2	HA18
AD1	HA19
AE1	HA20
AB5	TMS
AF1	HA21
AE2	HA22
AF2	HA23
AD3	HA24
AE3	HA25
AF3	HA26
AC4	HA27
AD4	HA28
AE4	HA29
AF4	HA30
AC5	HA31
AD5	NMI
AF5	INTR
AB6	PCICLKIN
AC6	IGERR#
AF6	AD31
AC7	AD30
AD7	AD29
AE7	AD28
AF7	AD27
AB8	AD26
AC8	AD25
AD8	AD24
AE8	AD23
AF8	AD22
AC9	AD21
AD9	AD20
AE9	AD19
AF9	AD18
AC10	AD17
AD10	AD16
AE10	AD15
AF10	AD14
AC11	AD13
AD11	AD12
AE11	AD11
AF11	AD10
AC12	AD9
AD12	AD8
AE12	AD7
AF12	AD6
AC13	AD5
AD13	AD4
AE13	AD3
AF13	AD2
AC14	AD1
AD14	AD0
AB9	FRAME#
AB11	IRDY#
AB12	TRDY#
AB14	PCICLK0
AE14	C/BE3#
AF14	C/BE2#
AC15	C/BE1#
AD15	C/BE0#
AE15	PLOCK#
AF15	DEVSEL#
AC16	STOP#
AE16	REQ2#
AF16	CLKRUN#
AC17	CPAR
AD17	SERR#
AE17	PERR#
AF17	REQ0#
AB18	REQ1#
AD18	REQ3#
AE18	IRQSER
AF18	IRQ1
AC19	IRQ3/IRQA
AD19	IRQ4/IRQB
AE19	IRQ5/IRQC
AF19	IRQ6/IRQD
AB20	CMD#
AC20	SEL#/ATB#
AD20	IRQ7/IRQE
AE20	IRQ8#
AF20	IRQ9/IRQF
AB22	IRQ10/IRQG
AC21	IRQ11/IRQH
AD21	IRQ12
AE21	IRQ14
AF21	IRQ15
AC22	SD15
AD22	SD14
AE22	SD13
AF22	SD12
AD23	SD11
AE23	SD10
AF23	SD9
AE24	SD8
AF24	SD7
AF25	SD6
AF26	SD5
AE25	SD4
AE26	SD3
AD24	SD2
AD25	SD1
AD26	SD0
AC25	RSTDRV
AC26	MRD#
AB23	MWR#
AB24	IOR#
AB25	IOW#
AB26	IOCHRDY
AA23	XD7
AA24	XD6
AA25	XD5
AA26	XD4
Y23	XD3
Y24	XD2
Y25	XD1
Y26	XD0
W23	IO16#
W24	M16#
W25	SBHE#
W26	SMRD#
V22	SMWR#
V23	SA23
V24	SA22
V25	SA21
V26	SA20
U23	SA19
U24	SA18
U25	SA17
U26	SA16
T22	SA15
T23	SA14
T24	SA13
T25	SA12
T26	SA11
R22	SA10
R23	SA9
R24	SA8
R25	SA7
R26	SA6
P23	SA5
P24	SA4
P25	SA3
P26	SA2
N22	SA1
N23	SA0
N24	RTCAS
N25	RTCRD#
N26	RTCWR#
M22	AEN
M23	TC
M24	DRQ0/DRQA
M25	DRQ1/DRQB
M26	DRQ2/DRQC
L23	DRQ3/DRQD
L24	DRQ5/DRQE
L25	DRQ6/DRQF
L26	DRQ7/DRQG
J24	ROMCS#
J25	RFSH#
J26	KBDCS#
H23	SPKOUT
H24	DBEW#
H25	DDRQ0
H26	PWRGD
K22	DACKA# **(NAND Output)**

8.0 Mechanical Package Outlines

Figure 8-1 432-Pin Ball Grid Array (BGA)



Dwg. No.:	AS432BGA-001		
Dwg. Rev.:	0	Unit:	MM

Appendix A. Compact ISA Specification

This document describes a new OPTi interface that will be used to interface the 82C852 PCMCIA Controller to OPTi system controller chipsets. This interface may also be used to interface OPTi peripheral products in the future. The interface is OPTi-proprietary, and may be licensed to others in the future.

A.1 Compact ISA Overview

The Compact ISA interface coexists with the standard ISA interface. Chips that support the Compact ISA interface enjoy a reduced ISA pin count because address signals and command information are strobed in on the SD[15:0] bus. ISA pins eliminated are:

- SA[23:0] (24 pins)
- IORD#, IOWR#, MRD#, MWR#, SMRD#, SMWR#, SBHE#, NOWS#, AEN, IO16#, M16# (11 pins)
- IRQ3, 4, 5, 6, 7, 10, 11, 12, 14, 15; DRQ/DACK#0, 1, 2, 3, 5, 6, 7, and TC (25 pins)

Compact ISA defines only two new signals, CMD# and SEL#/ATB#, for a total requirement of 22 pins. The pin count reduction over standard ISA is 58 pins. Compact ISA performance is comparable with that of 16-bit ISA bus peripheral devices. Moreover, Compact ISA does **not** interfere with standard ISA operations. The complete signal set of Compact ISA, referred to in the descriptions as CISA, is shown below.

Table A-1 Compact ISA (CISA) Interface Signals

Name	Type*	Description
MAD[15:0]	I/O	**Multiplexed Bus:** Used to transfer address, command, data, IRQ, DRQ, DACK information.
ATCLK	I	**Standard ISA Clock:** CISA device uses rising edge to clock in the first (address) phase.
ALE	I	**Standard ISA Address Latch Enable:** CISA peripheral device uses rising edge of ALE to latch the second (address and command) phase. CISA host uses falling edge of ALE to latch CMD# from peripheral device.
CMD#	I	**Command Indication:** Common to host and all devices on the CISA bus. The CISA host asserts CMD# during the data phase of the cycle to time the standard ISA command (IORD#/WR#, MRD#/WR#), and also asserts CMD# to acknowledge SEL#ATB#.
SEL#/ATB# (also CLKRUN#)	O Tristate	**Device Selected / ISA Bus Backoff Request:** Common to all peripheral devices on the CISA bus. When ALE is high, the CISA device asserts SEL# to indicate to the host that it is claiming the cycle. When ALE is low, the CISA device drives this signal to indicate that it has an interrupt and/or DMA request to make; the host acknowledges by asserting CMD#. After the host has preset the CISA device in a Stop Clock mode, the device can assert this signal asynchronously to restart the clock.
IOCHRDY	O Tristate	**Standard ISA Cycle Extension Request:** Used during memory and I/O cycles.
RSTDRV	I	**Standard ISA Bus Reset**

*Peripheral side

A.2 Compact ISA Cycle Definition

The MAD[15:0] lines contain different information for each phase of the bus cycle. The use of these lines varies according to whether a memory cycle or an I/O cycle is being run. Certain cycle definition bits are common to all cycles, as shown in Table A-2.

Retained Values

Entries marked "Same" retain the same value as in the previous phase, in order to reduce transitions where possible. However, the CISA peripheral device decode logic must **not** assume that these values will be stable. The bits may be reassigned in the future.

A.2.1 Memory Cycle

The MAD[15:0] bit meanings for each phase of a memory cycle are shown in Table A-3. The M/IO# bit is always 1 for memory cycles.

The general structure of Compact ISA memory cycles is shown in Figure A-1 and Figure A-2.

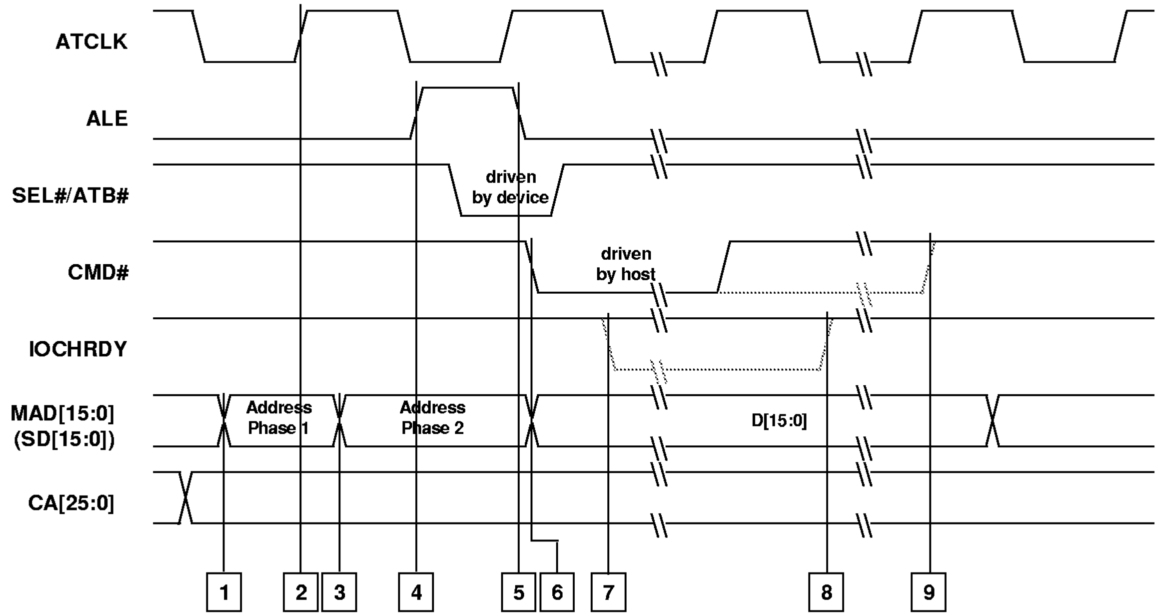
Table A-2 Common MAD Bit Usage

Signal	Phase 1	Phase 2
MAD0	M/IO# indication bit; used to determine the cycle type.	W/R# indication bit
MAD1	I/D# indication bit. It is always 0 if M/IO# = 1, and selects between I/O and DMA cycles if M/IO# = 0.	SBHE# indication bit
MAD2	Usage varies.	ISA# timing indication bit; described in the "Performance Control" section of this document.

Table A-3 MAD Bits During Memory Cycles

Phase	MAD15	MAD14	MAD13	MAD12	MAD11	MAD10	MAD9	MAD8	MAD7	MAD6	MAD5	MAD4	MAD3	MAD2	MAD1	MAD0
1	SA23	SA22	SA21	SA20	SA19	SA18	SA17	SA16	SA15	SA14	SA13	SA12	SA11	SA10	I/D# = 0	M/IO# = 1
2	SA9	SA8	SA7	SA6	SA5	SA4	SA3	SA2	SA1	SA0	Same	Same	Same	ISA#	SBHE#	W/R#
3	SD15	SD14	SD13	SD12	SD11	SD10	SD9	SD8	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0

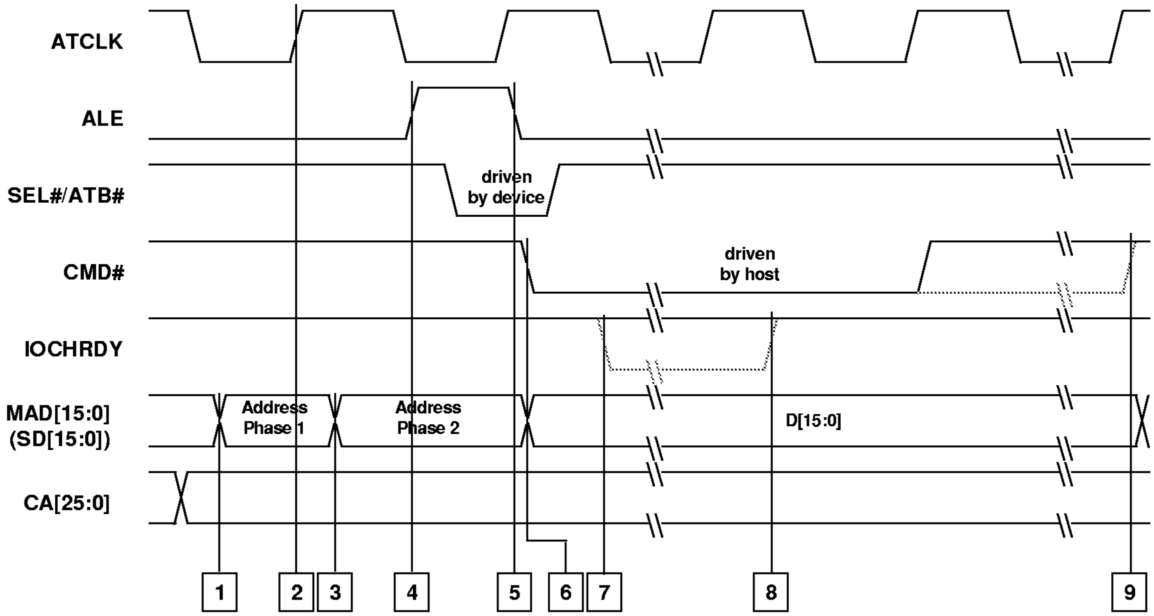
Figure A-1 Compact ISA Memory Cycle Operation, Fast CISA Timing*



*Cycle optionally extended by IOCHRDY shown in gray.

1. CISA host gets address from the CPU address lines and byte enable lines. The host then drives out A[23:10] + M/IO# on MAD[15:0] with M/IO# high (memory).
2. CISA peripheral device latches address and M/IO# on the rising edge of ATCLK and decodes the information.
3. Host drives out remaining address + Command on MAD[15:0].
4. Host asserts ALE. If cycle belongs to CISA peripheral device, it asserts SEL# and latches the address and command from MAD[15:0] on the rising edge of ALE. Device latches ISA# = 1 at this time.
5. Host and other CISA devices recognize the SEL# function of SEL#/ATB# by seeing ALE high when sampling SEL#/ATB# low on the rising edge of ATCLK. Host de-asserts ALE and stops driving address on this rising ATCLK edge.
6. For reads, the host tristates the MAD[15:0] buffers. For writes, it drives the write data onto MAD[15:0]. Host asserts CMD# synchronous to the rising edge of ATCLK and can optionally inhibit its MRD#/MWR# lines.
7. Cycle is 0 wait states as indicated by ISA# = 1. CISA peripheral device can bring IOCHRDY low asynchronously after CDM# goes active to extend the cycle.
8. Device brings IOCHRDY high synchronous to the falling edge of ATCLK to allow cycle completion.
9. Host de-asserts CMD# on the same rising edge where it samples IOCHRDY high.

Figure A-2 Compact ISA Memory Cycle Operation, Standard ISA Timing*



*Cycle optionally extended by IOCHRDY shown in gray.

1. CISA host gets address from the CPU address lines and byte enable lines. The host then drives out A]23:10] + M/IO# on MAD[15:0] with M/IO# high (memory).
2. CISA peripheral device latches address and M/IO# on the rising edge of ATCLK and decodes the information.
3. Host drives out remaining address + Command on MAD[15:0].
4. Host asserts ALE. If cycle belongs to CISA peripheral device, it asserts SEL# and latches the address and command from MAD[15:0] on the rising edge of ALE. Device latches ISA# = 0 at this time.
5. Host and other CISA devices recognize the SEL# function of SEL#/ATB# by seeing ALE high when sampling SEL#/ATB# low on the rising edge of ATCLK. Host de-asserts ALE and stops driving address on this rising ATCLK edge.
6. For reads, the host tristates the MAD[15:0] buffers. For writes, it drives the write data onto MAD[15:0]. Host asserts CMD# synchronous to the rising edge of ATCLK and can optionally inhibit its MRD#/MWR# lines.
7. Cycle is not zero wait states, as indicated by ISA# = 0. CISA peripheral device can bring IOCHRDY low asynchronously after CDM# goes active to extend the cycle further.
8. Device brings IOCHRDY high asynchronously to allow cycle completion.
9. Host de-asserts CMD# on the next rising edge of ATCLK after the rising edge ATCLK edge on which it samples IOCHRDY high.

A.2.2 I/O Cycle

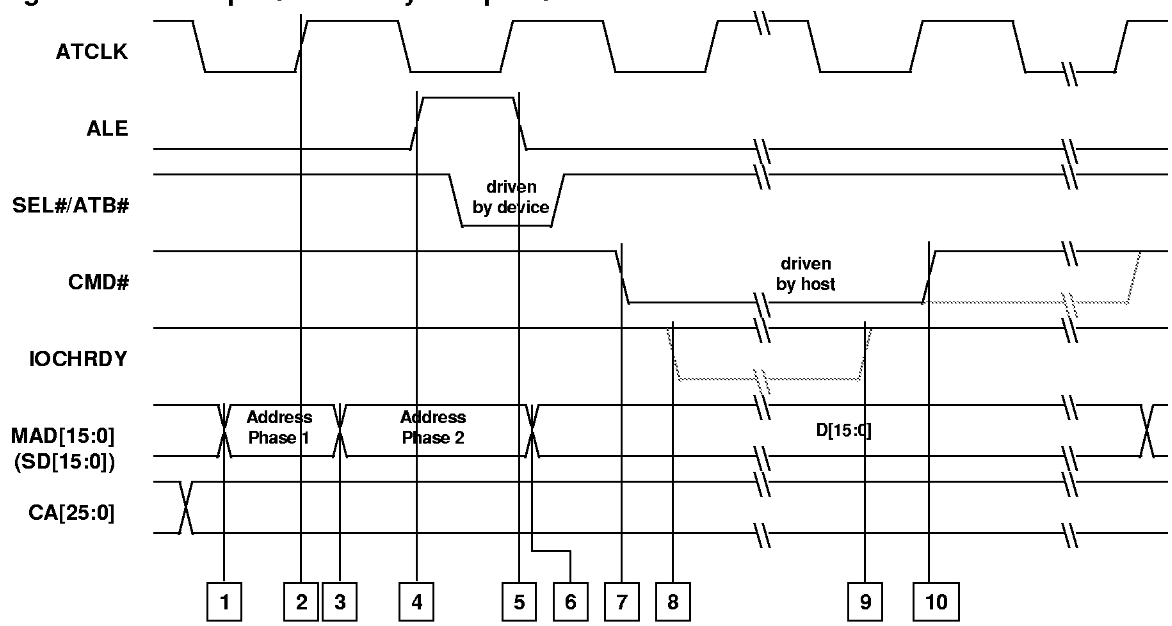
The MAD[15:0] bit meanings for each phase of an I/O cycle are shown below. The M/IO# bit is always 0, and the I/D# bit is always 1, for an I/O cycle.

The general structure of Compact ISA I/O cycles is shown in Figure A-3.

Table A-4 MAD Bits During I/O Cycles

Phase	MAD15	MAD14	MAD13	MAD12	MAD11	MAD10	MAD9	MAD8	MAD7	MAD6	MAD5	MAD4	MAD3	MAD2	MAD1	MAD0
1	SA9	SA8	SA7	SA6	SA5	SA4	SA3	SA2	SA15	SA14	SA13	SA12	SA11	SA10	I/D# = 1	M/IO# = 0
2	Same	Same	Same	Same	Same	Same	Same	Same	SA1	SA0	Same	Same	Same	ISA# = 0	SBHE#	W/R#
3	SD15	SD14	SD13	SD12	SD11	SD10	SD9	SD8	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0

Figure A-3 Compact ISA I/O Cycle Operation*



*Cycle optionally extended by IOCHRDY shown in gray.

1. CISA host gets address from the CPU address lines and byte enable lines. The host then drives out A[15:2] + I/D# = 1 + M/IO# = 0 (I/O cycle).
2. CISA peripheral device latches address and M/IO# on the rising edge of ATCLK and decodes the information.
3. Host drives out remaining address + Command on MAD[15:0].
4. Host asserts ALE. If cycle belongs to CISA peripheral device, it asserts SEL# and latches the address and command from MAD[15:0] on the rising edge of ALE.
5. Host and other CISA devices recognize the SEL# function of SEL#/ATB# by seeing ALE high when sampling SEL#/ATB# low on the rising edge of ATCLK. Host de-asserts ALE and stops driving address on this rising ATCLK edge.
6. For reads, the host tristates the MAD[15:0] buffers. For writes, it drives the write data onto MAD[15:0].
7. Host asserts CMD# synchronous to the falling edge of ATCLK to run the command and can optionally inhibit its IOR#/IOW# lines.
8. Cycle is never zero wait state. CISA peripheral device can bring IOCHRDY low asynchronously after CDM# goes active, using standard ISA setup timing, to extend the cycle further. Note that if CISA peripheral device provides a bridge to another device (a PCMCIA slot, for example), the device on the secondary bus must be able to return IOCHRDY soon enough to meet setup timing on the CISA interface.
9. Device brings IOCHRDY high asynchronously to allow cycle completion.
10. Host de-asserts CMD# on the next falling edge of ATCLK after the rising edge ATCLK edge on which it samples IOCHRDY high.

A.2.3 DMA on the CISA/ISA Bus

DMA operations are handled very specifically for CISA peripheral devices. Both CISA memory devices and CISA DMA devices can be involved in a DMA transfer, possibly at the same time. The CISA host must handle each situation.

The central consideration is that the CISA host must be able to distinguish between the DMA channels that are on the ISA bus and those that are on the CISA bus. This is a simple matter when the host also incorporates the DMA controller: because the host is responsible for latching the DRQ driveback information, it can determine on a cycle-by-cycle basis whether the DMA device being serviced is on CISA or on ISA according to whether it latched DRQ active for that channel from a CISA driveback cycle.

Because the host has this knowledge, the CISA DMA device does **not** need to assert SEL# on a DACK# cycle. The host already knows the cycle belongs to a CISA DMA device and does not need to see SEL# for the I/O portion of the cycle. This inhibition of SEL# is most important when a CISA memory device is responding to the memory portion of the cycle: the CISA memory device must respond as always with SEL#, and there would be contention (on deassertion) if the CISA DMA device asserted SEL# as well.

The host must foresee the following two situations.

- **DMA transfer between ISA DMA device and any memory device (system DRAM, ISA memory, or CISA memory) -** The host runs a standard CISA memory cycle (I/D# = 0, M/IO# = 1) along with the ISA memory-I/O cycle. If the selected memory is present on CISA, the device will respond to the access with SEL# as usual. The host **must** drop ALE if SEL# is returned.
- **DMA transfer between CISA DMA device and memory -** The host runs a CISA DACK# cycle (I/D# = 0, M/IO# = 0). If a CISA memory device claims this cycle it responds with SEL# as usual. The memory device can drive IOCHRDY low to extend the cycle if desired.

The CISA DACK# cycle is described below.

A.2.4 DACK# Cycle

The DACK# cycle is unique in that it has properties of a memory cycle but is directed to an I/O device. Basically, the DACK# cycle is a memory cycle whose address must be decoded by any CISA memory device on the bus. SBHE# and W/R# reference the memory device, not the I/O device; the I/O device must assume the opposite sense of W/R# for its portion of the cycle. Only the memory device responds with SEL#; the DMA (I/O) device never responds. The CMD# timing will be the wider pulse of MEMW#/IOR# or MEMR#/IOW#.

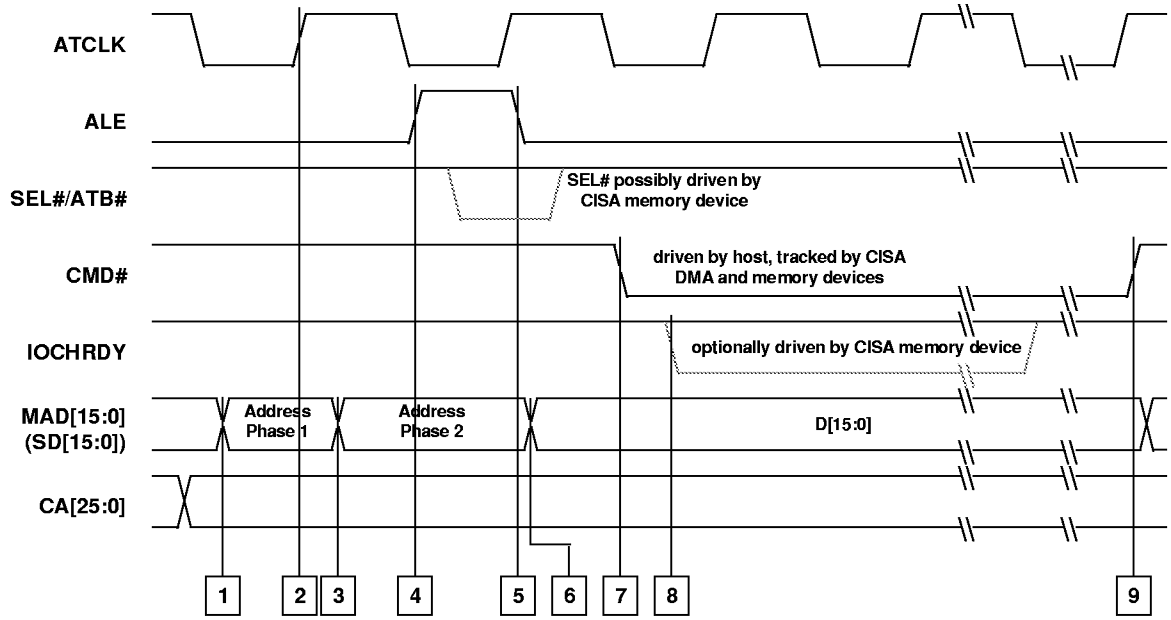
The MAD[15:0] bit meanings for each phase of a DMA acknowledge cycle are shown in Table A-5. The M/IO# bit is always 0, and the I/D# bit is always 0, for a DACK# cycle. DMX2-0 encode the number of the DACK#. For example, DMX2-0 = 010 indicate DACK2# active. TC is high if the DACK# is being returned with the Terminal Count indication. Note that there is no ISA# bit, since there is no fast cycle possible.

The general structure of Compact ISA DACK# cycles is shown in Figure A-4.

Table A-5 MAD Bits During DMA Acknowledge Cycles

Phase	MAD15	MAD14	MAD13	MAD12	MAD11	MAD10	MAD9	MAD8	MAD7	MAD6	MAD5	MAD4	MAD3	MAD2	MAD1	MAD0
1	SA23	SA22	SA21	SA20	SA19	SA18	SA17	SA16	SA15	SA14	SA13	SA12	SA11	SA10	I/D# = 0	M/IO# = 0
2	SA9	SA8	SA7	SA6	SA5	SA4	SA3	SA2	SA1	SA0	DMX2	DMX1	DMX0	TC	SBHE#	W/R#
3	SD15	SD14	SD13	SD12	SD11	SD10	SD9	SD8	SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0

Figure A-4 Compact ISA DACK# Cycle Operation



1. CISA host gets address form the CPU address lines and byte enable lines. The host then drives out A[23:0] + I/D# = 0 + M/IO# = 0 (DACK# cycle).
2. CISA DAM device, and possibly CISA memory device, latches address and cycle type information on the rising edge of TACLK and decodes the information.
3. Host drives out remaining command information on MAD[15:0].
4. Host asserts ALE. CIDA DMA device does not assert SEL# but latches the address and command from MAD[15:0] on the rising edge of ALE. Any CISA memory device present latches address and command, decodes them, and asserts SEL# if appropriate.
5. Host de-asserts ALE and stops driving address on this rising ATCLK edge. Note that in a normal ISA cycle the host would keep ALE high.
6. For DMA I/O read, the host tristates the MAD[15:0] buffers. For DMA I/O write, it drives the write data onto MAD[15:0].
7. Host asserts CMD# synchronous to the falling edge of ATCLK to run the command and is required to inhibit its IOR#/IOW# lines.
8. Only CISA memory devices can extend the cycle with IOCHRDY.
9. DACK# cycle is minimum 1.5 ATCLK. Host de-asserts CMD# synchronous to the rising edge of ATCLK.

A.2.5 Configuration Cycle

The CISA Configuration Cycle is a special cycle reserved for future expansion of CISA. The only configuration cycle currently defined is the Broadcast cycle; the only type of Broadcast cycle specified at this moment is the Stop Clock cycle.

The Stop Clock cycle indicates that the host will immediately put the CISA peripheral devices into a low-power mode in which they will no longer receive clocks. Therefore, the CISA peripheral device must enter into a state in which it can asynchronously signal that it needs the clocks restarted. CISA devices might need to generate an interrupt back to the system, which they cannot do if not receiving clocks.

The MAD[15:0] bit meanings for each phase of the Stop Clock configuration cycle are shown below.

In phase 1, the M/IO# bit is always 1, and the I/D# bit is always 1, for any configuration cycle. BRD is 1 to indicate a Broadcast cycle, and will always be zero for any other configuration cycle. The STP# bit is 0 to indicate a Stop Clock cycle, and will be 1 for all other cycles. Bits CC2:0 are the Clock Count bits that indicate to the CISA peripheral device how many rising clock edges to expect after CMD# goes high before the clock is actually stopped. The other bits of phase 1 are reserved and should not be decoded.

In phase 2, ISA# = 1 indicating that this will be a fast cycle. SBHE# = 0 to indicate 16 bits of data. W/R# = 1 because the Stop Clock Broadcast cycle is always a write cycle.

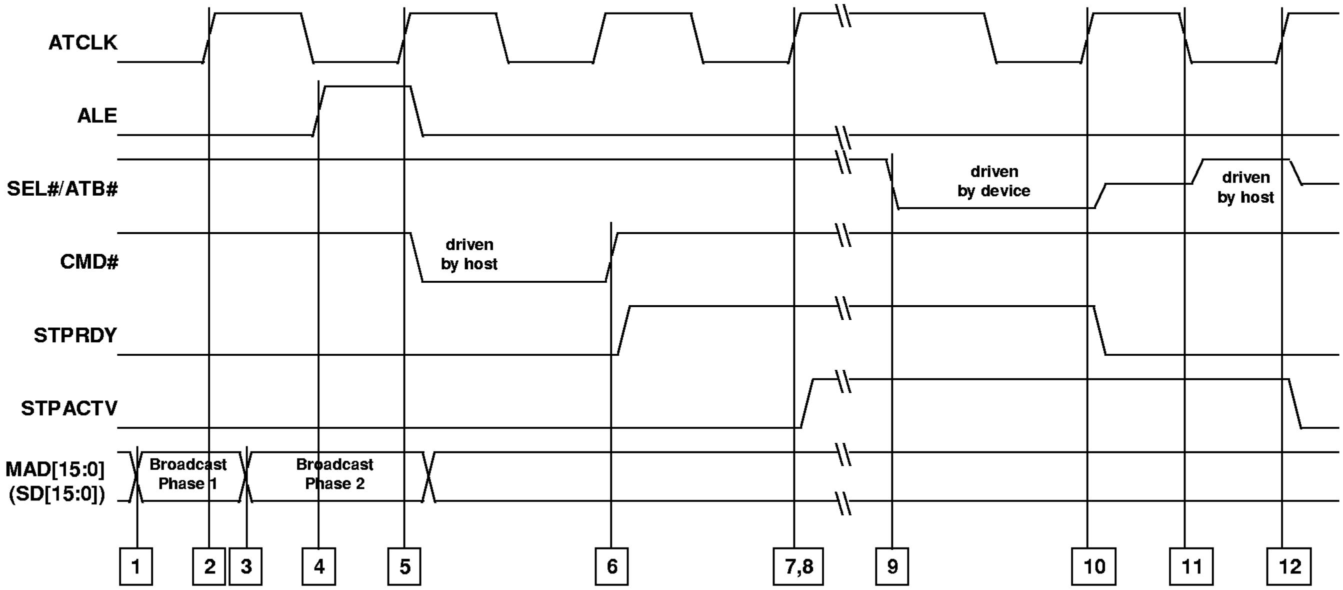
The data phase of the Stop Clock cycle contains no useful data and should not be latched.

The general structure of Compact ISA Broadcast cycles is shown in Figure A-5.

Table A-6 MAD Bits During Stop Clock Configuration Cycles

Phase	MAD15	MAD14	MAD13	MAD12	MAD11	MAD10	MAD9	MAD8	MAD7	MAD6	MAD5	MAD4	MAD3	MAD2	MAD1	MAD0
1	BRD = 1	STP# = 0	CC2	CC1	CC0	Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	I/D# = 1	M/IO# = 1
2	Same	Same	Same	Same	Same	Same	Same	Same	Same	Same	Same	Same	Same	ISA# = 1	SBHE#	W/R# = 1
3	Same	Same	Same	Same	Same	Same	Same	Same	Same	Same	Same	Same	Same	Same	Same	Same

Figure A-5 Compact ISA Configuration Cycle Operation



This example describes the Broadcast configuration cycle

1. CISA host initiates the Configuration cycle; it is not generate form ISA commands. The host drives out BRD = 1 + I/D# = 1 + M/IO# = 0 (Broadcast configuration cycle).
2. CISA peripheral latches the command data on the rising edge of ATCLK and decodes the information.
3. Host drives out Clock Count, Stop Clock cycle indicator, and remaining command information on MAD[15:0].
4. Host asserts ALE. CISA devices latch clock count. CISA peripheral devices must NOT respond with SEL#.
5. Host asserts CMD# synchronous to the rising edge of ATCLK to run the command. The Broadcast configuration cycle is always zero wait states so it completes in one ATCLK.
6. After the host de-asserts CMD#, the CISA peripheral device is internally in STPRDY state.
7. After the number of clocks specified by CC[2:0], the host stops the clock in its high state. In the example, CC[2:0] = 001 (the minimum allowed) so the host will stop the clock on the next rising ATCLK edge. Each additional count requires the host to wait one more clock.
8. The CISA peripheral device is also counting clocks while in STPRDY state. On the specified ATCLK edge the device is in STPACTV state. In STPACTV state, the CISA peripheral device gives SEL#/ATB# a third meaning: CLKRUN#. The device can assert CLKRUN# asynchronously at any time while in this mode to get the host to restart its clocks.
9. CISA peripheral device asserts CLKRUN# (SEL#/ATB#) on receipt of an interrupt to restart the clocks.
10. On next rising ATCLK clock edge, CISA peripheral device de-asserts CLKRUN# (SEL#/ATB#) but must not drive it high. Device has left STPRDY state but is still in STPACTV state and cannot initiate or respond to any cycle.
11. On next falling ATCLK edge, the host drives SEL#/ATB# high for ½ ATCLK.
12. On next rising ATCLK edge, the host stops driving SEL#/ATB#. The CISA peripheral device leaves STPACTV state on this clock edge and can either generate an interrupt driveback cycle or can respond to cycles from the host.

A.3 Interrupt and DMA Request Drive-Back

Compact ISA provides the signal SEL#/ATB# to give the CISA peripheral device limited ownership of the bus. The SEL#/ATB# signal acts as ATB# (AT backoff) when asserted with ALE low. When the device asserts ATB# to the host, the host inhibits further AT bus operations and asserts the CMD# line to the CISA peripheral device to acknowledge that the device now owns the bus. The peripheral device can only drive two types of information onto the bus: interrupt requests and DMA requests.

Figure A-6 illustrates the synchronous IRQ/DRQ driveback cycle.

A.3.1 Interrupt Requests

To drive interrupt requests, the CISA peripheral device drives the MAD[15:0] lines low for each IRQ line it wishes to assert. The host side IRQ generation circuitry samples ATB# and CMD# active on the rising ATCLK edge and latches the IRQ information on MAD[15:0].

The IRQ generation circuitry, whether external or built into the host, determines how to treat IRQ information. For pulse-type interrupts it could latch the IRQs and enable tristate buffers to drive the lines low for 1-3 ATCLKs, for example.

A.3.2 DMA Requests

The CISA device must always precede the DRQ drive-back cycle with an IRQ drive-back cycle, even if no IRQs have changed state.

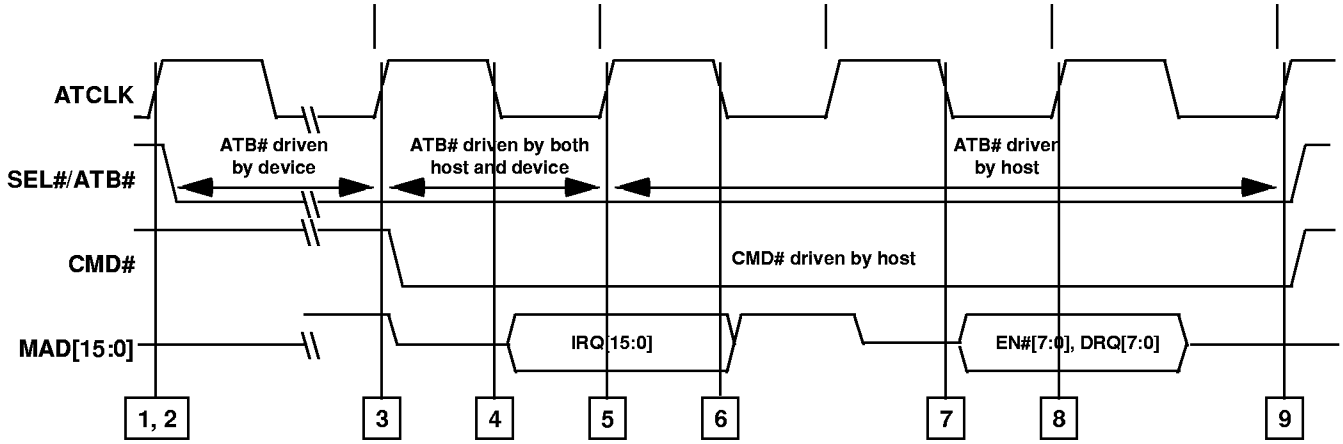
To make DMA requests, the CISA peripheral device drives the MAD[15:8] lines low for each DRQ it wishes to change. The device then sets the state of each MAD[7:0] line to correspond to the DRQ state desired. The host side DRQ generation circuitry samples ATB# and CMD# active on the next rising ATCLK edge after the edge on which IRQs were sampled, and latches the DRQ information on MAD[7:0] for the channels selected on MAD[15:8].

The desired DMA request line states are latched by the host and will remain in that state until cleared by another DRQ drive-back cycle. This scheme allows both DMA single transfer and DMA block transfer modes to be used. The CISA peripheral device must assert SEL#/ATB# immediately any time a DRQ line changes state (assuming the current cycle is finished). The CISA host, in turn, must immediately deassert all DRQ inputs to its DMA controller until the drive-back cycle is complete.

Table A-7 IRQ/DRQ Drive Back Cycle

Phase	MAD15	MAD14	MAD13	MAD12	MAD11	MAD10	MAD9	MAD8	MAD7	MAD6	MAD5	MAD4	MAD3	MAD2	MAD1	MAD0
IRQ	IRQ15	IRQ14	IRQ13	IRQ12	IRQ11	IRQ10	IRQ9	IRQ8	IRQ7	IRQ6	IRQ5	IRQ4	IRQ3	IRQ2	IRQ1	IRQ0
DRQ	EN7#	EN6#	EN5#	Rsvd	EN3#	EN2#	EN1#	EN0#	DRQ7	DRQ6	DRQ5	Rsvd	DRQ3	DRQ2	DRQ1	DRQ0

Figure A-6 Compact ISA Interrupt and DMA Request Drive-Back Cycle



1. CISA Peripheral device must sample SEL#/ATB# and CMD# high, and ALE low, on TWO consecutive rising edges of ATCLK.
2. CISA peripheral device asserts ATB# on rising edge of ATCLK to request AT backoff. If host was starting a cycle and was about to assert ALE on the next falling edge of ATCLK, it must abort the cycle and retry it later. Even if host is busy and cannot respond to drive back request immediately, it inhibits initiation of all I/O and DMA operations (EOI to PCI is blocked, for example).
3. As soon as AT bus operations have been completed and bus is available, host drives MAD[15:0] high for ½ ATCLK from a falling edge of ATCLK, then asserts CMD# after the net rising edge of ATCLK. The host drives ATB# low at this time.
4. CISA peripheral device(s) can drive interrupt data onto bus on next falling edge of ATCLK, driving low only those lines with IRQ activity and not actively driving high the other lines. In this way, multiple CISA devices can drive the lines in parallel.
5. Host IRQ generation circuitry uses rising edge of ATCLK, qualified by ATB# and CMD# low, to latch IRQs. The CISA device stops driving ATB# at this time. The host controls ATB# throughout the rest of the cycle.
6. CISA peripheral device drives any MAD[15:-0] lines it was driving low high for ½ ATCLK, then tristates the lines for ½ ATCLK.
7. CISA peripheral device drives DRQ information onto MAD[7:0] and at the same time drives low the corresponding lines MAD[15:8] to indicate which DRQ channels have a status change to be transferred.
8. Host DRQ generation circuitry uses next rising edge of ATLCK, qualified by ATB# and CMD# low AND previous IRQ cycle, to latch DRQs. The host DRQ generation circuitry ORs the DRQs with other system DRQs.
9. Host de-asserts CMD# and ATB# on rising edge of ATCLK.

A.4 Performance Control

Compact ISA performance is comparable with that of 16-bit ISA bus peripheral devices. In its simplest implementation, the CMD# signal is simply an AND of MRD#, MWR#, IOR#, and IOW# from the standard AT controller state machine.

Memory cycles are always assumed to be **zero wait state**. The CISA host detects a NOWS# command every time SEL# is generated. The CISA peripheral device can use its IOCHRDY line to extend the cycle and override the NOWS# status. All of this functionality is consistent with standard ISA operation.

I/O cycles cannot be made zero-wait-state cycles on the ISA bus, so by default are not zero-wait-state cycles on the CISA bus. However, performance improvement is possible if the CMD# duration is shortened to one ATCLK. Future PCMCIA I/O devices may be able to complete their cycles this quickly, for example. For zero-wait-state CISA I/O operation, the cycle timing would have to change from the standard ISA timing. The host can implement fast CISA timing as an option. However, all CISA slave devices are **required** to be able to accept fast CISA timing.

Fast CISA timing on the host side is defined as follows. If the CISA host is driving CMD# as derived from the logical AND of ISA command lines IOR#, IOW#, MRD#, and MWR#, it sets ISA# = 0 to indicate that the CISA peripheral device must assume ISA timing. If the host is capable of performing fast CISA cycles, it can set ISA# = 1. In this case, the CISA peripheral device must deassert IOCHRDY early to lengthen cycles.

Fast CISA timing on the device side is defined as follows. If the CISA host drives the ISA# bit low, the CISA peripheral device assumes normal ISA timing for CMD# and IOCHRDY. If the CISA host drives ISA# high, the CISA peripheral device must drop IOCHRDY low immediately upon receiving CMD# to lengthen the cycle; this is different from ISA timing.

The CISA peripheral device will have a programmable option to determine how IOCHRDY is deasserted. By default, the device might drop IOCHRDY on every cycle. For the example of a PCMCIA controller on the CISA bus, only when a fast PCMCIA card is inserted (as indicated in the CIS header of the card) would Card Services be allowed to enable the fast CISA timing option on the CISA peripheral device side.

A.5 Compatibility and Host Responsibilities

Compact ISA does **not** interfere with standard ISA operations or limit compatibility. This statement can be made with only the following restrictions:

- No device can drive the SD bus between ISA cycles. Devices capable of driving the SD bus must stay tristated at this time.
- ATCLK can be stopped only after a Stop Clock Broadcast configuration cycle. Slower-than-standard clock speeds are allowed if interrupt latency is not an issue.
- ISA bus masters cannot access CISA devices. Standard ISA masters are simply ignored by CISA devices since these masters cannot generate CMD# and so cannot run a CISA cycle. ISA bus masters can still take bus control and communicate with other ISA peripherals. CISA interrupt latency may be an issue if a bus master prevents the CISA host from responding to ATB# for an interrupt driveback cycle.
- No CISA bus master capability is currently defined. However, the presence of the SEL#/ATB# signal and its AT backoff feature leave open the possibility of future bus master capabilities.
- On receipt of an ATB# request, the CISA host must immediately inhibit all system DRQ activity (possibly by deasserting all DRQs to the DMA controller) until the drive-back cycle is complete. Otherwise, unwanted DMA cycles could occur.

A.6 Shared Speaker Signal Support (Optional)

Compact ISA provides a new scheme for the digital speaker output signal common to PCs and PCMCIA controllers. This scheme allows all digital audio outputs to be tied together without the XOR logic usually required.

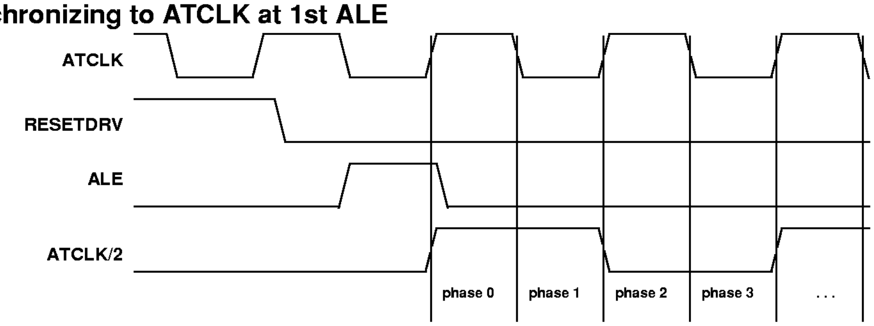
The standard specification for the speaker data output is a signal driven in both the low-to-high and high-to-low directions. The output cannot simply be respecified as open-collector, since there is no guarantee that software will leave the speaker output line from the system chipset in a high or tristated condition. If it leaves the signal driven low, no other open-collector devices connected on the line could toggle the signal. Moreover, open collector outputs tend to consume excessive power.

Compact ISA provides an efficient solution to the problem as described in the following sections.

A.6.1 Initial Synchronization

All CISA slave devices must tristate their SPKR outputs at hard reset time and remain tristated until individually enabled. On the first ALE generated by the host, all participating CISA devices will synchronize to ATCLK and derive the signal ATCLK/2 that is in phase as shown in Figure A-7. Four distinct phases, 0 through 3, are the result. CISA slave SPKR outputs are still tristated at this point.

Figure A-7 Synchronizing to ATCLK at 1st ALE



A.6.2 SPKR Sharing During Active Mode

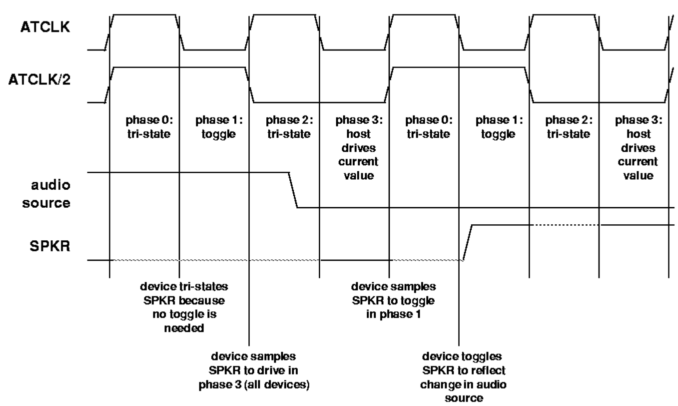
The activities performed in each phase by the CISA host and the CISA slaves are as given in Table A-8.

Figure A-8 illustrates the SPKR handling requirements.

Table A-8 SPKR Sharing During Active Mode

Phase	Slave	Host
On the rising ATCLK edge starting phase 0	Sample the state of SPKR.	Sample the state of SPKR. Tristate SPKR output.
During phase 0:	Maintain SPKR output tristated (as it was from previous phase).	Maintain SPKR output tristated.
On the falling ATCLK edge starting phase 1:	Sample digital audio source input.	
During phase 1:	If digital audio source input sampled on ATCLK edge has changed state since the previous phase 1 in which it was sampled, toggle SPKR. SPKR is toggled by driving the opposite of the SPKR value sampled in phase 0 onto the SPKR output.	
On the rising ATCLK edge starting phase 2:	Tristate SPKR output.	Tristate SPKR output. Sample the state of SPKR.
During phase 2:	Slave and host: Maintain SPKR output tristated.	
On the falling ATCLK edge starting phase 3:	No activity on this edge.	Drive SPKR pin to the value of SPKR sampled in phase 2.
During phase 3:	Maintain SPKR output tristated (as it was from previous phase).	Maintain SPKR output driven.

Figure A-8 Shared SPKROUT Signal Management



A.6.3 SPKR Sharing During Stop Clock Mode

During Stop Clock mode, CISA devices handle SPKR as follows.

Slave: Tristate SPKR. Referring to Figure A-5, the exact period during which CISA slaves keep SPKR tristated is defined as the period during which both STPACTV and STPRDY are high.

Host: Drive or tristate SPKR. It is recommended that the host drive SPKR low.

Note that even while CISA slave devices are in Stop Clock mode, they must remain synchronized to the correct phase of ATCLK. They do **not** resynchronize on the next ALE.

A.6.4 Audio Output Circuit Recommendations

The SPKR output must **never** be connected directly to a speaker or other low-impedance transducer. The shared SPKR implementation depends on an R-C time constant large enough that the signal will never change its level any appreciable amount across a period of 1.5 ATCLKs, the maximum number of clocks for which no device will be driving the SPKR line.

Three ATCLKs last for approximately 188ns. The R-C time constant of the design must be significantly larger than this value. Connecting an 8 ohm speaker directly would cause the line to begin a transition when it was tristated. Therefore, either capacitive coupling or an amplifier circuit with a high-impedance input is recommended.

A.7 Automatic Voltage Threshold Detection

Compact ISA devices are intended to work on either a traditional 5.0V ISA bus or on a local 3.3V ISA bus. Compact ISA designs are very power-conscious, so using external strap options on each CISA device to select the input buffer threshold may not be the best option.

Therefore, the Compact ISA host is required to use the ALE pin at reset to indicate the ISA bus voltage to CISA slaves. The correspondence is as follows.

- For a 5.0V ISA bus, the host must assert the ALE signal **high** when RSTDRV goes high, and must keep it asserted for at least 1/2 ATCLK and at most 1 ATCLK after RSTDRV goes low.
- For a 3.3V ISA bus, the host must keep the ALE signal **low** when RSTDRV goes high, and must maintain ALE low for at least 1/2 ATCLK after RSTDRV goes low.

This performance is **required** for CISA hosts, but CISA slave devices are not required to use the feature.

OPTi

Sales Information

HEADQUARTERS:

OPTi Inc.
888 Tasman Drive
Milpitas, CA 95035
tel: 408-486-8000
fax: 408-486-8011

SALES OFFICES:

Japan

OPTi Japan KK
Murata Building 6F, 2-22-7
Ohhashi Meguro-ku
Tokyo 153, Japan
tel: 81-3-5454-0178
fax: 81-3-5454-0168

Taiwan

OPTi Inc.
9F, No 303, Sec 4, Hsin Yih Road
Taipei, Taiwan, ROC
tel: 886-2-325-8520
fax: 886-2-325-6520

United Kingdom & Europe

OPTi Inc.
30 Windmill Avenue
Bicester, Oxon OX6 7DY
U.K.
tel: + 44-1-869-369-161
fax: Same

United States

OPTi Inc.
4400 N. Federal Highway, Ste. #120
Boca Raton, FL 33431
tel: 561-395-4555
fax: 561-395-4554

OPTi Inc.
20405 State Highway 249, Ste. #220
Houston, TX 77070
tel: 281-257-1856
fax: 281-257-1825

REPRESENTATIVES:

United States

Alabama/Mississippi

Concord Component Reps
190 Line Quarry Rd., Ste. #102
Madison, AL 35758
tel: 205-772-8883
fax: 205-772-8262

Florida

Engineered Solutions Ind., Inc.
1000 E. Atlantic Blvd., Ste. #202
Pompano Beach, FL 33060
tel: 305-784-0078
fax: 305-781-7722

Georgia

Concord Component Reps
6825 Jimmy Carter Blvd., Ste. #1303
Norcross, GA 30071
tel: 770-416-9597
fax: 770-441-0790

Illinois

Micro-Tex, Inc.
1870 North Roselle Rd., Ste. #107
Schaumburg, IL 60195-3100
tel: 708-885-8200
fax: 708-885-8210

Massachusetts

S-J Associates, Inc.
267 Boston Road
Corporate Place, Ste. #3
N. Billerica, MA 01862
tel: 508-670-8899
fax: 508-670-8711

Michigan

Jay Marketing
44752 Helm Street., Ste. A
Plymouth, MI 48170
tel: 313-459-1200
fax: 313-459-1697

New Jersey

S-J Associates, Inc.
131-D Gaither Dr.
Mt. Laurel, NJ 08054
tel: 609-866-1234
fax: 609-866-8627

New York

S-J Associates, Inc.
265 Sunrise Highway
Rockville Centre, NY 11570
tel: 516-536-4242
fax: 516-536-9638

S-J Associates, Inc.
735 Victor-Pittsford
Victor, NY 14564
tel: 716-924-1720

North & South Carolina

Concord Component Reps
10608 Dunhill Terrace
Raleigh, NC 27615
tel: 919-846-3441
fax: 919-846-3401

Ohio/W. Pennsylvania

Lyons Corp.
4812 Fredrick Rd., Ste. #101
Dayton, OH 45414
tel: 513-278-0714
fax: 513-278-3609

Lyons Corp.
4615 W. Streetsboro
Richfield, OH 44286
tel: 216-659-9224
fax: 216-659-9227

Lyons Corp.
248 N. State St.
Westerville, OH 43081
tel: 614-895-1447
fax: Same

Texas

Axxis Technology Marketing, Inc.
701 Brazos, Suite 500
Austin, TX 78701
tel: 512-320-9130
fax: 512-320-5730

Axxis Technology Marketing, Inc.
6804 Ashmont Drive
Plano, TX 75023
tel: 214-491-3577
fax: 214-491-2508

Virginia

S-J Associates, Inc.
900 S. Washington St., Ste. #307
Falls Church, VA 22046
tel: 703-533-2233
fax: 703-533-2236

Wisconsin

Micro-Tex, Inc.
22660 Broadway, Ste. #4A
Waukesha, WI 53186
tel: 414-542-5352
fax: 414-542-7934

International

Australia

Braemac Pty. Ltd.
Unit 6, 111 Moore St., Leichhardt
Sydney, 2040 Australia
tel: 61-2-550-6600
fax: 61-2-550-6377

China

Legend Electronic Components. Ltd.
Unit 413, Hong Kong Industrial
Technology Centre
72 Tat Chee Avenue
Kowloon Tong, Hong Kong
tel: 852-2776-7708
fax: 852-2652-2301

France

Tekelec Airtronic, France
5, Rue Carle Vernet
92315 Sevres Cedex
France
tel: 33-1-46-23-24-25
fax: 33-1-45-07-21-91

Germany

Kamaka
Rheinsrasse 22
76870 Kandel
Germany
tel: 49-7275-958211
fax: 49-7275-958220

India

Spectra Innovation
Unit S-822 Manipal Centre
47 Dickenson Road
Bangalore 560-042
Kamataka, India
tel: 91-80-558-8323/3977
fax: 91-80-558-6872

Israel

Ralco Components (1994) Ltd.
11 Benyamini St.
67443 Tel Aviv
Israel
tel: 972-3-6954126
fax: 972-3-6951743

Korea

Woo Young Tech Co., Ltd.
5th Floor Koami Bldg
13-31 Yoido-Dong
Youngduengpo-Ku
Seoul, Korea 150-010
tel: 02-369-7099
fax: 02-369-7091

Singapore

Instep Microsolutions Pte Ltd.
629 Aljunied Road #05-15
Cititech Industrial Building
Singapore 1438
tel: 65-741-7507
65-741-7530
fax: 65-741-1478

South America

Uniao Digital
Rua Guido Caloi
Bloco B, Piso 3
Sao Paulo-SP, CEP 05802-140 Brazil
tel: 55-11-5514-3355
fax: 55-11-5514-1088

Switzerland

Datacomp AG
Silbernstrasse 10
8953 Dietikon
Switzerland
tel: 41-1-740-5140
fax: 41-1-741-3423

United Kingdom

Spectrum
2 Grange Mews,
Station Road
Launton, Bicester
Oxfordshire,OX6 0DX
UK
tel: 44-1869-325174
fax: 44-1869-325175

MMD
3 Bennet Court,
Bennet Road
Reading
Berkshire, RG2 0QX
UK
tel: 44 1734 313232
fax: 44 1734 313255

(2/97a)

OPTi Inc.
888 Tasman Drive
Milpitas, CA 95035
Tel: (408) 486-8000
Fax: (408) 486-8001
WWW: http://www.opti.com/