82378 SYSTEM I/O (SIO)

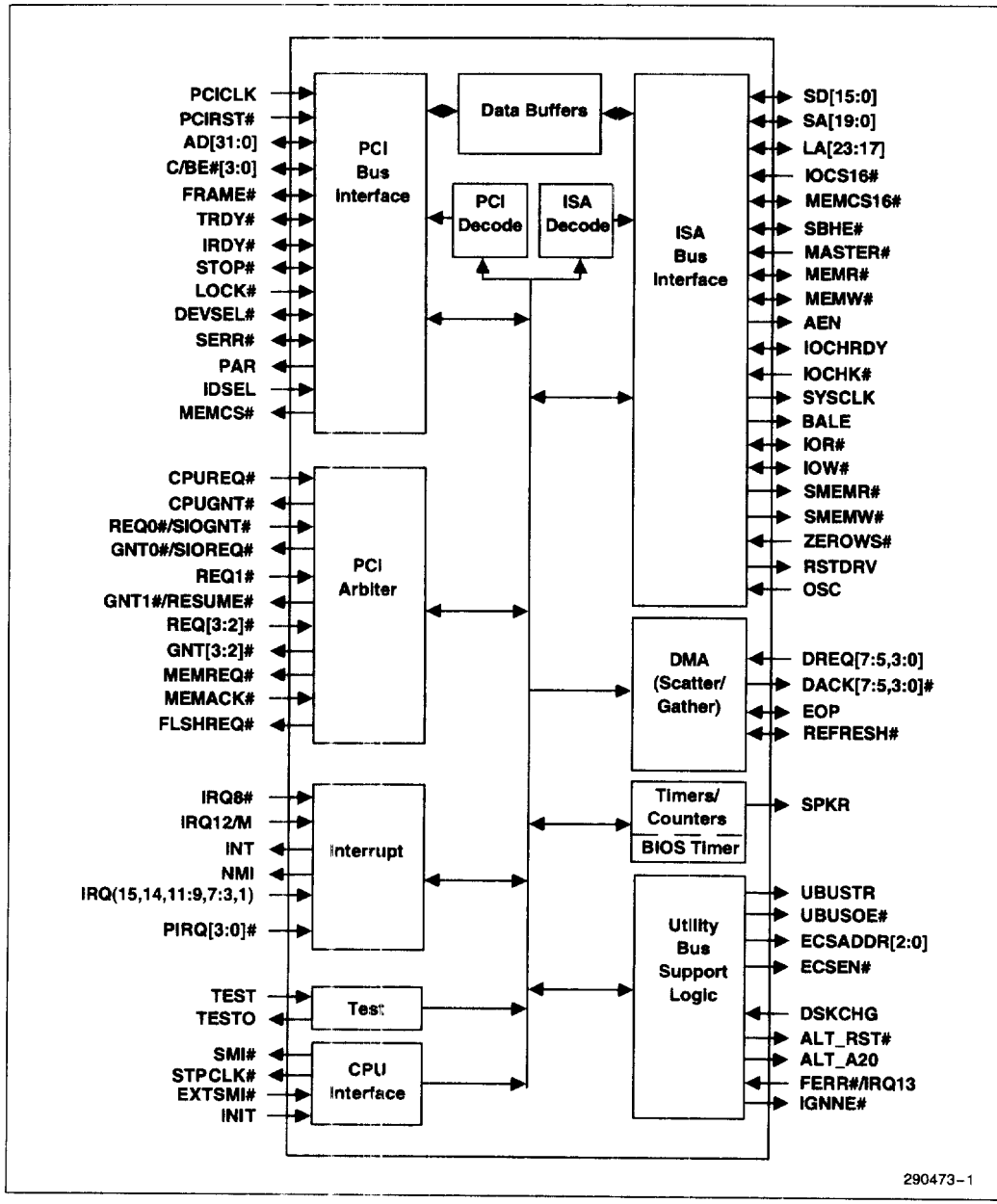
- **Provides the Bridge Between the PCI Bus and ISA Bus**
- **100% PCI and ISA Compatible**
  - **— PCI and ISA Master/Slave Interface**
  - **— Directly Drives 10 PCI Loads and 6 ISA Slots**
  - **— Supports PCI at 25 MHz and 33 MHz**
  - **— Supports ISA from 6 MHz to 8.33 MHz**
- **Enhanced DMA Functions**
  - **— Scatter/Gather**
  - **— Fast DMA Type A, B and F**
  - **— Compatible DMA Transfers**
  - **— 32-bit Addressability**
  - **— Seven Independently Programmable Channels**
  - **— Functionality of Two 82C37A DMA Controllers**
- **Integrated Data Buffers to Improve Performance**
  - **— 8-Byte DMA/ISA Master Line Buffer**
  - **— 32-bit Posted Memory Write Buffer to ISA**
- **Integrated 16-bit BIOS Timer**
- **Non-Maskable Interrupts (NMI)**
  - **— PCI System Errors**
  - **— ISA Parity Errors**
- **Arbitration for ISA Devices**
  - **— ISA Masters**
  - **— DMA and Refresh**
- **Four Dedicated PCI Interrupts**
  - **— Level Sensitive**
  - **— Can be Mapped to Any Unused Interrupt**
- **Arbitration for PCI Devices**
  - **— Six PCI Masters Supported**
  - **— Fixed, Rotating, or a Combination of the Two**
- **Utility Bus (X-Bus) Peripheral Support**
  - **— Provides Chip Select Decode**
  - **— Controls Lower X-Bus Data Byte Transceiver**
- **Integrates the Functionality of One 82C54 Timer**
  - **— System Timer**
  - **— Refresh Request**
  - **— Speaker Tone Output**
- **Integrates the Functionality of Two 82C59 Interrupt Controllers**
  - **— 14 Interrupts Supported**
  - **— Edge/Level Selectable Interrupts: Each Interrupt Individually Programmable**
- **Complete Support for SL Enhanced Intel486 CPU's**
  - **— SMI# Generation Based on System Hardware Events**
  - **— STPCLK# Generation to Power Down the CPU**

1

The 82378 System I/O (SIO) component provides the bridge between the PCI bus and the ISA expansion bus. The SIO also integrates many of the common I/O functions found in today's ISA based PC systems. The SIO ncorporates the logic for a PCI interface (master and slave), ISA interface (master and slave), enhanced seven channel DMA controller that supports fast DMA transfers and Scatter/Gather, data buffers to isolate the PCI bus from the ISA bus and to enhance performance, PCI and ISA arbitration, 14 level interrupt controller, a 16-bit BIOS timer, three programmable timer/counters, and Non-Maskable Interrupt (NMI) Control Logic. The SIO also provides decode for peripheral devices such as the Flash BIOS, Real Time Clock, Keyboard/Mouse Controller, Floppy Controller, two Serial Ports, one Parallel Port, and IDE Hard Disk Drive.

The 82378 also supports several Advanced Power Management features such as SMI#, APM Register, Fast On and Fast Off Event Timers, Clock Throttling, and support for an external SMI# Interrupt. The 82378 also supports a total of 6 PCI Masters, and can support up to 4 PCI Interrupts.

December 1994
Order Number: 290473-004



SIO Component Block Diagram

82378 SYSTEM I/O (SIO)

CONTENTS

CONTENTS

CONTENTS

1

CONTENTS

PAGE

CONTENTS

CONTENTS

CONTENTS

1.0 ARCHITECTURAL OVERVIEW

The major functions of the SIO component are broken up into blocks as shown in the SIO Component Block Diagram. A description of each block is provided below.

PCI Bus Interface

The PCI Bus Interface provides the interface between the SIO and the PCI bus. The SIO provides both a master and slave interface to the PCI bus. As a PCI master, the SIO runs cycles on behalf of DMA, ISA masters, and the internal data buffer management logic when buffer flushing is required. The SIO will burst a maximum of two Dwords when reading from PCI memory, and one Dword when writing to PCI memory. The SIO does not generate PCI I/O cycles as a master. As a PCI slave, the SIO accepts cycles initiated by PCI masters targeted for the SIO's internal register set or the ISA bus. The SIO will accept a maximum of one data transaction before terminating the transaction. This supports the Incremental Latency Mechanism as defined in the Peripheral Component Interconnect (PCI) Specification.

As a master, the SIO generates address and command signal (C/BE#) parity for read and write cycles, and data parity for write cycles. As a slave, the SIO generates data parity for read cycles. Parity checking is not supported. The SIO also provides support for system error reporting by generating a Non-Maskable-Interrupt (NMI) when SERR# is driven active.

The SIO, as a resource, can be locked by any PCI master. In the context of locked cycles, the entire SIO subsystem (including the ISA bus) is considered a single resource.

The SIO directly supports the PCI Interface running at either 25 MHz or 33 MHz. If a frequency of less than 33 MHz is required (not including 25 MHz), a SYSCLK divisor value (as indicated in the ISA Clock Divisor Register) must be selected that guarantees that the ISA bus frequency does not violate the 6 MHz to 8.33 MHz SYSCLK range.

PCI Arbiter

The PCI arbiter provides support for six PCI masters; the Host Bridge, SIO, and four PCI masters. The arbiter can be programmed for a purely rotating scheme, fixed, or a combination of the two. The Arbiter can also be programmed to support bus parking. This gives the Host Bridge default access to the PCI bus when no other device is requesting service. The arbiter can be disabled if an external arbiter is used.

PCI Decode/ISA Decode

The SIO contains two address decoders; one to decode PCI initiated cycles and one to decode ISA master and DMA initiated cycles. Two decoders are used to allow the PCI and ISA buses to run concurrently.

The SIO is also programmable to provide address decode on behalf of the Host Bridge. When programmed, the SIO monitors the PCI and ISA address buses, and generates a memory chip select signal (MEMCS#) indicating that the current cycle is targeted for system memory residing behind the Host Bridge. This feature can be disabled through software.

Data Buffers

To isolate the slower ISA bus from the PCI bus, the SIO provides two types of data buffers. One Dword deep posted write buffer is provided for the posting of PCI initiated memory write cycles to the ISA bus. The second buffer is a bi-directional, 8-byte line buffer used for ISA master and DMA accesses to the PCI bus. All DMA and ISA master read and write cycles go through the 8-byte line buffer.

The data buffers also provide the data assembly or disassembly when needed for transactions between the PCI and ISA buses.

Buffering is programmable and can be enabled or disabled through software.

ISA Bus Interface

The SIO incorporates a fully ISA-bus compatible master and slave interface. The SIO directly drives six ISA slots without external data or address buffering. The ISA interface also provides byte swap logic, I/O recovery support, wait-state generation, and SYSCLK generation. The SIO supports ISA bus frequencies from 6 MHz to 8.33 MHz.

As an ISA master, the SIO generates cycles on behalf of DMA, Refresh, and PCI master initiated cycles. The SIO supports compressed cycles when accessing ISA slaves (i.e. ZEROWS# asserted). As an ISA slave, the SIO accepts ISA master accesses targeted for the SIO's internal register set or ISA master memory cycles targeted for the PCI bus. The SIO does not support ISA master initiated I/O cycles targeted for the PCI bus.

The SIO also monitors ISA master to ISA slave cycles to generate SMEMR# or SMEMW#, and to support data byte swapping, if necessary.

DMA

The DMA controller incorporates the functionality of two 82C37 DMA controllers with seven independently programmable channels. Each channel can be programmed for 8- or 16-bit DMA device size, and ISA-compatible or fast DMA type "A", type "B", or type "F" timings. Full 32-bit addressing is supported as an extension of the ISA-compatible specification. The DMA controller is also responsible for generating ISA refresh cycles.

The DMA controller supports an enhanced feature called Scatter/Gather. This feature provides the capability of transferring multiple buffers between memory and I/O without CPU intervention. In Scatter/Gather mode, the DMA can read the memory address and word count from an array of buffer descriptors, located in system memory, called the Scatter/Gather Descriptor (SGD) Table. This allows the DMA controller to sustain DMA transfers until all of the buffers in the SGD table are read.

Timer Block

The timer block contains three counters that are equivalent in function to those found in one 82C54 programmable interval timer. These three counters are combined to provide the System Timer function, Refresh Request, and speaker tone. The three counters use the 14.31818 MHz OSC input for a clock source.

In addition to the three counters, the SIO provides a programmable 16-bit BIOS timer. This timer can be used by BIOS software to implement timing loops. The timer uses the ISA system clock (SYSCLK) divided by 8 as a clock source. An 8:1 ratio between the SYSCLK and the BIOS timer clock is always maintained. The accuracy of the BIOS timer is ±1 ms.

Utility Bus (X-Bus) Logic

The SIO provides four encoded chip selects that are decoded externally to provide chip selects for Flash BIOS, Real Time Clock, Keyboard/Mouse Controller, Floppy Controller, two Serial Ports, one Parallel Port, and an IDE Hard Disk Drive. The SIO provides the control for the buffer that isolates the lower eight bits of the Utility Bus from the lower 8 bits of the ISA bus.

In addition to providing the encoded chip selects and Utility Bus buffer control, the SIO also provides Port 92 functions (Alternate Reset and Alternate A20), Coprocessor error reporting, the Floppy DSKCHG function, and a mouse interrupt input.

Interrupt Controller Block

The SIO provides an ISA compatible interrupt controller that incorporates the functionality of two 82C59 interrupt controllers. The two interrupt controllers are cascaded so that 14 external and 2 internal interrupts are possible.

Test

The test block provides the interface to the test circuitry within the SIO. The test input can be used to tri-state all of the SIO outputs.

2.0 PIN ASSIGNMENT

The SIO package is a 208-pin Quad Flatpack (QFP). The package signals are listed in Table 1. The following notations are used to describe pin types.

Signal Type	Description
I	**Input** is a standard input-only signal.
O	**Totem Pole Output** is a standard active driver.
OD	**Open Drain** Input/Output
IO	**Input/Output** is a bidirectional, tri-state pin.
s/t/s	**Sustained Tri-State** is an active low tri-state signal owned and driven by one and only one agent at a time. The agent that drives a s/t/s pin low must drive it high for at least one clock before letting it float. A new agent can not start driving a s/t/s signal any sooner than one clock after the previous owner tri-states it. A pull-up sustains the inactive state until another agent drives it and is provided by the central resource.
t/s/o	**Tri-State Output**

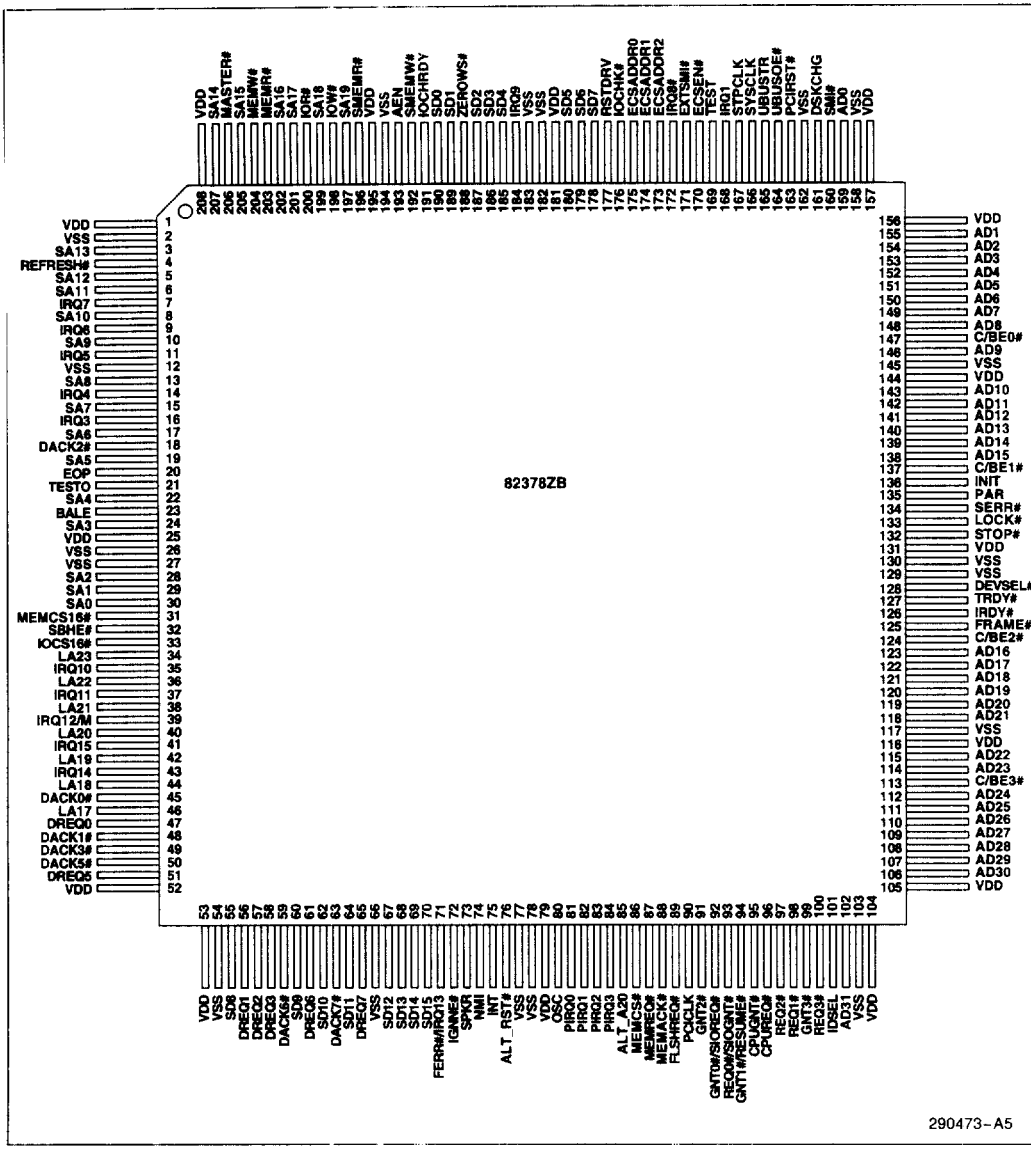


Figure 1. SIO Package Pinout Diagram

Table 1. Alphabetical Pin Assignment

Pin Name	Pin #	Type
AD0	159	I/O
AD1	155	I/O
AD2	154	I/O
AD3	153	I/O
AD4	152	I/O
AD5	151	I/O
AD6	150	I/O
AD7	149	I/O
AD8	148	I/O
AD9	146	I/O
AD10	143	I/O
AD11	142	I/O
AD12	141	I/O
AD13	140	I/O
AD14	139	I/O
AD15	138	I/O
AD16	123	I/O
AD17	122	I/O
AD18	121	I/O
AD19	120	I/O
AD20	119	I/O
AD21	118	I/O
AD22	115	I/O
AD23	114	I/O
AD24	112	I/O
AD25	111	I/O
AD26	109	I/O
AD27	109	I/O
AD28	108	I/O
AD29	107	I/O
AD30	106	I/O
AD31	102	I/O
AEN	193	O
ALT_A20	85	O
ALT_RST#	76	O

Pin Name	Pin #	Type
BALE	23	O
C/BE0#	147	I/O
C/BE1#	137	I/O
C/BE2#	124	I/O
C/BE3#	113	I/O
CPUGNT#	95	t/s/o
CPUREQ#	96	I
DACK0#	45	O
DACK1#	48	O
DACK2#	18	O
DACK3#	49	O
DACK5#	50	O
DACK6#	59	O
DACK7#	63	O
DEVSEL#	128	I/O (s/t/s)
DREQ0	47	I
DREQ1	56	I
DREQ2	57	I
DREQ3	58	I
DREQ5	51	I
DREQ6	61	I
DREQ7	65	I
DSKCHG	161	I
ECSADDR0	175	O
ECSADDR1	174	O
ECSADDR2	173	O
ECSEN#	170	O
EOP	20	I/O
EXTSMI#	171	I
FERR#/IRQ13	71	I
FLSHREQ#	89	t/s/o
FRAME#	125	I/O (s/t/s)
GNT0#/SIOREQ#	92	t/s/o
GNT1#/RESUME#	94	t/s/o
GNT2#	91	t/s/o

Table 1. Alphabetical Pin Assignment (Continued)

Pin Name	Pin #	Type
GNT3#	99	t/s/o
IDSEL	101	I
IGNNE#	72	O
INIT	136	I
INT	75	O
IOCHK#	176	I
IOCHRDY	191	I/O
IOCS16#	33	I
IOR#	200	I/O
IOW#	198	I/O
IRDY#	126	I/O (s/t/s)
IRQ1	168	I
IRQ3	16	I
IRQ4	14	I
IRQ5	11	I
IRQ6	9	I
IRQ7	7	I
IRQ8#	172	I
IRQ9	184	I
IRQ10	35	I
IRQ11	37	I
IRQ12/M	39	I
IRQ14	43	I
IRQ15	41	I
LA17	46	I/O
LA18	44	I/O
LA19	42	I/O
LA20	40	I/O
LA21	38	I/O
LA22	36	I/O
LA23	34	I/O
LOCK#	133	I (s/t/s)
MASTER#	206	I
MEMACK#	88	I
MEMCS#	86	O

Pin Name	Pin #	Type
MEMCS16#	31	I/O (o/d)
MEMR#	203	I/O
MEMREQ#	87	t/s/o
MEMW#	204	I/O
NMI	74	O
OSC	80	I
PAR	135	O
PCICLK	90	I
PCIRST#	163	I
PIRQ0#	81	I
PIRQ1#	82	I
PIRQ2#	83	I
PIRQ3#	84	I
REFRESH#	4	I/O
REQ0#/SIOGNT#	93	I
REQ1#	98	I
REQ2#	97	I
REQ3#	100	I
RSTDRV	177	O
SA0	30	I/O
SA1	29	I/O
SA2	28	I/O
SA3	24	I/O
SA4	22	I/O
SA5	19	I/O
SA6	17	I/O
SA7	15	I/O
SA8	13	I/O
SA9	10	I/O
SA10	8	I/O
SA11	6	I/O
SA12	5	I/O
SA13	3	I/O
SA14	207	I/O
SA15	205	I/O

Table 1. Alphabetical Pin Assignment (Continued)

Pin Name	Pin #	Type
SA16	202	I/O
SA17	201	I/O
SA18	199	I/O
SA19	197	I/O
SBHE#	32	I/O
SD0	190	I/O
SD1	189	I/O
SD2	187	I/O
SD3	186	I/O
SD4	185	I/O
SD5	180	I/O
SD6	179	I/O
SD7	178	I/O
SD8	55	I/O
SD9	60	I/O
SD10	62	I/O
SD11	64	I/O
SD12	67	I/O
SD13	68	I/O
SD14	69	I/O
SD15	70	I/O
SERR#	134	I
SMEMR#	196	O
SMEMW#	192	O
SMI#	160	O
SPKR	73	O
STOP#	132	I/O (s/t/s)
STPCLK#	167	O
SYSCLK	166	O
TEST	169	I
TEST0	21	O
TRDY#	127	I/O (s/t/s)
UBUSOE#	164	O
UBUSTR	165	O

Pin Name	Pin #	Type
ZEROWS#	188	I
V_{DD}	1	V
V_{DD}	79	V
V_{DD}	104	V
V_{DD}	105	V
V_{DD}	116	V
V_{DD}	131	V
V_{DD}	144	V
V_{DD}	156	V
V_{DD}	157	V
V_{DD}	181	V
V_{DD}	25	V
V_{DD}	52	V
V_{DD}	53	V
V_{DD}	195	V
V_{DD}	208	V
V_{SS}	2	V
V_{SS}	12	V
V_{SS}	26	V
V_{SS}	27	V
V_{SS}	54	V
V_{SS}	66	V
V_{SS}	77	V
V_{SS}	78	V
V_{SS}	103	V
V_{SS}	117	V
V_{SS}	129	V
V_{SS}	130	V
V_{SS}	145	V
V_{SS}	158	V
V_{SS}	162	V
V_{SS}	182	V
V_{SS}	183	V
V_{SS}	194	V

Table 2. Numerical Pin Assignment

Pin Name	Pin #	Type
V_{DD}	1	V
V_{SS}	2	V
SA13	3	I/O
REFRESH#	4	I/O
SA12	5	I/O
SA11	6	I/O
IRQ7	7	I
SA10	8	I/O
IRQ6	9	I
SA9	10	I/O
IRQ5	11	I
V_{SS}	12	V
SA8	13	I/O
IRQ4	14	I
SA7	15	I/O
IRQ3	16	I
SA6	17	I/O
DACK2#	18	O
SA5	19	I/O
EOP	20	I/O
TEST0	21	O
SA4	22	I/O
BALE	23	O
SA3	24	I/O
V_{DD}	25	V
V_{SS}	26	V
V_{SS}	27	V
SA2	28	I/O
SA1	29	I/O
SA0	30	I/O
MEMCS16#	31	I/O (o/d)
SBHE#	32	I/O
IOCS16#	33	I
LA23	34	I/O
IRQ10	35	I

Pin Name	Pin #	Type
LA22	36	I/O
IRQ11	37	I
LA21	38	I/O
IRQ12/M	39	I
LA20	40	I/O
IRQ15	41	I
LA19	42	I/O
IRQ14	43	I
LA18	44	I/O
DACK0#	45	O
LA17	46	I/O
DREQ0	47	I
DACK1#	48	O
DACK3#	49	O
DACK5#	50	O
DREQ5	51	I
V_{DD}	52	V
V_{DD}	53	V
V_{SS}	54	V
SD8	55	I/O
DREQ1	56	I
DREQ2	57	I
DREQ3	58	I
DACK6#	59	O
SD9	60	I/O
DREQ6	61	I
SD10	62	I/O
DACK7#	63	O
SD11	64	I/O
DREQ7	65	I
V_{SS}	66	V
SD12	67	I/O
SD13	68	I/O
SD14	69	I/O
SD15	70	I/O

Table 2. Numerical Pin Assignment (Continued)

Pin Name	Pin #	Type
FERR#/IRQ13	71	I
IGNNE#	72	O
SPKR	73	O
NMI	74	O
INT	75	O
ALT_RST#	76	O
V_{SS}	77	V
V_{SS}	78	V
V_{DD}	79	V
OSC	80	I
PIRQ0#	81	I
PIRQ1#	82	I
PIRQ2#	83	I
PIRQ3#	84	I
ALT_A20	85	O
MEMCS#	86	O
MEMREQ#	87	t/s/o
MEMACK#	88	I
FLSHREQ#	89	t/s/o
PCICLK	90	I
GNT2#	91	t/s/o
GNT0#/SIOREQ#	92	t/s/o
REQ0#/SIOGNT#	93	I
GNT1#/RESUME#	94	t/s/o
CPUGNT#	95	t/s/o
CPUREQ#	96	I
REQ2#	97	I
REQ1#	98	I
GNT3#	99	I
REQ3#	100	I
IDSEL	101	I
AD31	102	I/O
V_{SS}	103	V
V_{DD}	104	V
V_{DD}	105	V

Pin Name	Pin #	Type
AD30	106	I/O
AD29	107	I/O
AD28	108	I/O
AD27	109	I/O
AD26	110	I/O
AD25	111	I/O
AD24	112	I/O
C/BE3#	113	I/O
AD23	114	I/O
AD22	115	I/O
V_{DD}	116	V
V_{SS}	117	V
AD21	118	I/O
AD20	119	I/O
AD19	120	I/O
AD18	121	I/O
AD17	122	I/O
AD16	123	I/O
C/BE2#	124	I/O
FRAME#	125	I/O (s/t/s)
IRDY#	126	I/O (s/t/s)
TRDY#	127	I/O (s/t/s)
DEVSEL#	128	I/O (s/t/s)
V_{SS}	129	V
V_{SS}	130	V
V_{DD}	131	V
STOP#	132	I/O (s/t/s)
LOCK#	133	I (s/t/s)
SERR#	134	I
PAR	135	O
INIT	136	I
C/BE1#	137	I/O
AD15	138	I/O
AD14	139	I/O
AD13	140	I/O

Table 2. Numerical Pin Assignment (Continued)

Pin Name	Pin #	Type
AD12	141	I/O
AD11	142	I/O
AD10	143	I/O
V_{DD}	144	V
V_{SS}	145	V
AD9	146	I/O
C/BE0#	147	I/O
AD8	148	I/O
AD7	149	I/O
AD6	150	I/O
AD5	151	I/O
AD4	152	I/O
AD3	153	I/O
AD2	154	I/O
AD1	155	I/O
V_{DD}	156	V
V_{DD}	157	V
V_{SS}	158	V
AD0	159	I/O
SMI#	160	O
DSKCHG	161	I
V_{SS}	162	V
PCIRST#	163	I
UBUSOE#	164	O
UBUSTR	165	O
SYSCLK	166	O
STPCLK#	167	O
IRQ1	168	I
TEST	169	I
ECSEN#	170	O
EXTSMI#	171	I
IRQ8#	172	I
ECSADDR2	173	O
ECSADDR1	174	O

Pin Name	Pin #	Type
ECSADDR0	175	O
IOCHK#	176	I
RSTDRV	177	O
SD7	178	I/O
SD6	179	I/O
SD5	180	I/O
V_{DD}	181	V
V_{SS}	182	V
V_{SS}	183	V
IRQ9	184	I
SD4	185	I/O
SD3	186	I/O
SD2	187	I/O
ZEROWS#	188	I
SD1	189	I/O
SD0	190	I/O
IOCHRDY	191	I/O
SMEMW#	192	O
AEN	193	O
V_{SS}	194	V
V_{DD}	195	V
SMEMR#	196	O
SA19	197	I/O
IOW#	198	I/O
SA18	199	I/O
IOR#	200	I/O
SA17	201	I/O
SA16	202	I/O
MEMR#	203	I/O
MEMW#	204	I/O
SA15	205	I/O
MASTER#	206	I
SA14	207	I/O
V_{DD}	208	V

3.0 SIGNAL DESCRIPTION

This section contains a detailed description of each signal. The signals are arranged in functional groups according to the interface.

Note that 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.

3.1 PCI Bus Interface Signals

Signal Name	Type	Description
PCICLK	I	**PCI CLOCK:** PCICLK provides timing for all transactions on the PCI Bus. All other PCI signals are sampled on the rising edge of PCICLK, and all timing parameters are defined with respect to this edge. Frequencies supported by the SIO include 25 MHz and 33 MHz.
PCIRST#	I	**PCI RESET:** PCIRST# forces the SIO to a known state. AD[31:0], C/BE[3:0]#, and PAR are always driven low by the SIO synchronously from the leading edge of PCIRST#. The SIO always tri-states these signals from the trailing edge of PCIRST#. If the internal arbiter is enabled (CPUREQ# sampled high on the trailing edge of PCIRST#), the SIO will drive these signals low again (synchronously 2–5 PCICLKs later) until the bus is given to another master. If the internal arbiter is disabled (CPUREQ# sampled low on the trailing edge of PCIRST#), these signals remain tri-stated until the SIO is required to drive them valid as a master or slave. FRAME#, IRDY#, TRDY#, STOP#, DEVSEL#, MEMREQ#, FLSHREQ#, CPUGNT#, GNT0#/SIOREQ#, and GNT1#/RESUME# are tri-stated from the leading edge of PCIRST#. FRAME#, IRDY#, TRDY#, STOP#, and DEVSEL# remain tri-stated until driven by the SIO as either a master or a slave. MEMREQ#, FLSHREQ#, CPUGNT#, GNT0#/SIOREQ#, and GNT1#/RESUME# are tri-stated until driven by the SIO. After PCIRST#, MEMREQ# and FLSHREQ# are driven inactive asynchronously from PCIRST# inactive. CPUGNT#, GNT0#/SIOREQ#, and GNT1#/RESUME# are driven based on the arbitration scheme and the asserted REQx#'s. All registers are set to their default values. PCIRST# may be asynchronous to PCICLK when asserted or negated. Although asynchronous, negation must be a clean, bounce-free edge. Note that PCIRST# must be asserted for more than 1 μs.

3.1 PCI Bus Interface Signals (Continued)

Signal Name	Type	Description
AD[31:0]	I/O	**PCI ADDRESS/DATA.** AD[31:0] is a multiplexed address and data bus. During the first clock of a transaction, AD[31:0] contain a physical byte address (32 bits). During subsequent clocks, AD[31:0] contain data. A SIO Bus transaction consists of an address phase followed by one or more data phases. Little-endian byte ordering is used. AD[7:0] define the least significant byte (LSB) and AD[31:24] the most significant byte (MSB). When the SIO is a target, AD[31:0] are inputs during the address phase of a transaction. During the following data phase(s), the SIO may be asked to supply data on AD[31:0] for a PCI read, or accept data for a PCI write. As a master, the SIO drives a valid address on AD[31:2] during the address phase, and drives write or latches read data on AD[31:0] during the data phase. The SIO always drives AD[1:0] low as a master. AD[31:0] are always driven low by the SIO synchronously from the leading edge of PCIRST#. The SIO always tri-states AD[31:0] from the trailing edge of PCIRST#. If the internal arbiter is enabled (CPUREQ# sampled high on the trailing edge of PCIRST#), the SIO drives AD[31:0] low again (synchronously 2–5 PCICLKs later) until the bus is given to another master. If the internal arbiter is disabled (CPUREQ# sampled low on the trailing edge of PCIRST#), AD[31:0] remain tri-stated until the SIO is required to drive them valid as a master or slave. When the internal arbiter is enabled, the SIO acts as the central resource responsible for driving the AD[31:0] signals when no one is granted the PCI Bus and the bus is idle. When the internal arbiter is disabled, the SIO does not drive AD[31:0] as the central resource. The SIO is always responsible for driving AD[31:0] when it is granted the bus (SIOGNT# and idle bus) and as appropriate when it is the master of a transaction.
C/BE[3:0]#	I/O	**BUS COMMAND AND BYTE ENABLES:** The command and byte enable signals are multiplexed on the same PCI pins. During the address phase of a transaction, C/BE[3:0]# define the bus command. During the data phase C/BE[3:0]# are used as Byte Enables. The Byte Enables determine which byte lanes carry meaningful data. C/BE#[0] applies to byte 0, C/BE[1] to byte 1, C/BE[2] to byte 2, and C/BE#[3] to byte 3. The SIO drives C/BE[3:0]# as an initiator of a PCI Bus cycle and monitors C/BE[3:0]# as a Target. C/BE[3:0]# are always driven low by the SIO synchronously from the leading edge of PCIRST#. The SIO always tri-states C/BE[3:0]# from the trailing edge of PCIRST#. If the internal arbiter is enabled (CPUREQ# sampled high on the trailing edge of PCIRST#), the SIO drives C/BE[3:0]# low again (synchronously 2-5 PCICLKs later) until the bus is given to another master. If the internal arbiter is disabled (CPUREQ# sampled low on the trailing edge of PCIRST#), C/BE[3:0]# remain tri-stated until the SIO is required to drive them valid as a master or slave. When the internal arbiter is enabled, the SIO acts as the central resource responsible for driving the C/BE[3:0]# signals when no one is granted the PCI Bus and the bus is idle. When the internal arbiter is disabled, the SIO does not drive C/BE[3:0]# as the central resource. The SIO is always responsible for driving C/BE[3:0]# when it is granted the bus (SIOGNT# and idle bus) and as appropriate when it is the master of a transaction.

3.1 PCI Bus Interface Signals (Continued)

Signal Name	Type	Description
FRAME#	I/O (s/t/s)	**CYCLE FRAME:** FRAME# is driven by the current master to indicate the beginning and duration of an access. FRAME# is asserted to indicate a bus transaction is beginning. While FRAME# is asserted data transfers continue. When FRAME# is negated the transaction is in the final data phase. FRAME# is an input to the SIO when the SIO is the target. FRAME# is an output when the SIO is the initiator. FRAME# is tri-stated from the leading edge of PCIRST#. FRAME# remains tri-stated until driven by the SIO as either a master or a slave.
TRDY#	I/O (s/t/s)	**TARGET READY:** TRDY# indicates the SIO's ability to complete the current data phase of the transaction. TRDY# is used in conjunction with IRDY#. A data phase is completed when both TRDY# and IRDY# are sampled asserted. During a read, TRDY# indicates that the SIO, as a target, has placed valid data on AD[31:0]. During a write, it indicates the SIO, as a target is prepared to latch data. TRDY is an input to the SIO when the SIO is the initiator and an output when the SIO is a target. TRDY# is tri-stated from the leading edge of PCIRST#. TRDY# remains tri-stated until driven by the SIO as either a master or a slave.
IRDY#	I/O (s/t/s)	**INITIATOR READY:** IRDY# indicates the SIO's ability, as an initiator, to complete the current data phase of the transaction. It is used in conjunction with TRDY#. A data phase is completed on any clock that both IRDY# and TRDY# are sampled asserted. During a write, IRDY# indicates the SIO has valid data present on AD[31:0]. During a read, it indicates the SIO is prepared to latch data. IRDY is an input to the SIO when the SIO is the target and an output when the SIO is an initiator. IRDY# is tri-stated from the leading edge of PCIRST#. IRDY# remains tri-stated until driven by the SIO as either a master or a slave.
STOP#	I/O (s/t/s)	**STOP:** STOP# indicates that the SIO, as a target, is requesting a master to stop the current transaction. As a master, STOP# causes the SIO to stop the current transaction. STOP# is an output when the SIO is a target and an input when the SIO is an initiator. STOP# is tri-stated from the leading edge of PCIRST#. STOP# remains tri-stated until driven by the SIO as either a master or a slave.
LOCK#	I	**LOCK:** LOCK# indicates an atomic operation that may require multiple transactions to complete. LOCK# is always an input to the SIO. When the SIO is the target of a transaction and samples LOCK# negated during the address phase of a transaction, the SIO considers itself a locked resource until it samples LOCK# and FRAME# negated. When other masters attempt accesses while the SIO is locked, the SIO responds with a retry termination. LOCK# is tri-stated during reset.
IDSEL	I	**INITIALIZATION DEVICE SELECT:** IDSEL is used as a chip select during configuration read and write transactions. The SIO samples IDSEL during the address phase of a transaction. If IDSEL is sampled active, and the bus command is a configuration read or write, the SIO responds by asserting DEVSEL# on the next cycle.
DEVSEL#	I/O (s/t/s)	**DEVICE SELECT:** The SIO asserts DEVSEL# to claim a PCI transaction through positive or subtractive decoding. As an output, the SIO asserts DEVSEL# when it samples IDSEL active in configuration cycles to SIO configuration registers. The SIO also asserts DEVSEL# when an internal SIO address is decoded or when the SIO subtractively decodes a cycle. As an input, DEVSEL# indicates the response to a SIO master-initiated transaction. The SIO also samples this signal for all PCI transactions to decide to subtractively decode the cycle. DEVSEL# is tri-stated from the leading edge of PCIRST#. DEVSEL# remains tri-stated until driven by the SIO as either a master or a slave.

3.1 PCI Bus Interface Signals (Continued)

Signal Name	Type	Description
PIRQ[3:0]#	I	**PCI INTERRUPT REQUEST:** PIRQ#s are used to generate asynchronous interrupts to the CPU via the Programmable Interrupt Controllers (82C59s) integrated in the SIO. These signals are defined as level sensitive and are asserted low. The PIRQx# interrupts can be steered into any unused IRQ interrupt. The PIRQx# Route Control Register determines which IRQ interrupt each PCI interrupt is steered into. These pins include a weak internal pull-up resistor.
PAR	O	**CALCULATED PARITY SIGNAL:** PAR is "even" parity and is calculated on 36 bits—AD[31:0] plus C/BE[3:0]#. "Even" parity means that the number of "1"s within the 36 bits plus PAR are counted and the sum is always even. PAR is always calculated on 36 bits regardless of the valid byte enables. PAR is generated for address and data phases and is only guaranteed to be valid one PCI clock after the corresponding address or data phase. PAR is driven and tri-stated identically to the AD[31:0] lines except that PAR is delayed by exactly one PCI clock. PAR is an output during the address phase (delayed one clock) for all SIO master transactions. It is also an output during the data phase (delayed one clock) when the SIO is the master of a PCI write transaction, and when it is the target of a read transaction. PAR is always driven low by the SIO synchronously from the leading edge of PCIRST#. The SIO always tri-states PAR from the trailing edge of PCIRST#. If the internal arbiter is enabled (CPUREQ# sampled high on the trailing edge of PCIRST#), the SIO drives PAR low again (synchronously 2-5 PCICLKs later) until the bus is given to another master. If the internal arbiter is disabled (CPUREQ# sampled low on the trailing edge of PCIRST#), PAR remains tri-stated until the SIO is required to drive them valid as a master or slave. When the internal arbiter is enabled, the SIO acts as the central resource responsible for driving PAR when no device is granted the PCI Bus and the bus is idle. When the internal arbiter is disabled, the SIO does not drive PAR as the central resource. The SIO is always responsible for driving PAR when it is granted the bus (SIOGNT# and idle bus) and as appropriate when it is the master of a transaction.
SERR#	I	**SYSTEM ERROR:** SERR# can be pulsed active by any PCI device that detects a system error condition. Upon sampling SERR# active, the SIO generates a non-maskable interrupt (NMI) to the CPU.

3.2 PCI Arbiter Signals

Signal Name	Type	Description
CPUREQ#	I	**CPU REQUEST:** This signal provides the following functions: 1. If CPUREQ# is sampled high on the trailing edge of PCIRST#, the internal arbiter is enabled. If CPUREQ# is sampled low on the trailing edge of PCIRST#, the internal arbiter is disabled. This requires that the host bridge drive CPUREQ# high during PCIRST#. 2. If the SIO's internal arbiter is enabled, this pin is configured as CPUREQ#. An active low assertion indicates that the CPU initiator desires the use of the PCI Bus. If the internal arbiter is disabled, this pin is meaningless after reset. This pin has a weak internal pull-up resistor.
REQ0#/ SIOGNT#	I	**REQUEST 0/SIO GRANT:** If the SIO's internal arbiter is enabled, this pin is configured as REQ0#. An active low assertion indicates that Initiator0 desires the use of the PCI Bus. If the internal arbiter is disabled, this pin is configured as SIOGNT#. When asserted, SIOGNT# indicates that the external PCI arbiter has granted use of the bus to the SIO. This pin has a weak internal pull-up resistor.
REQ1#	I	**REQUEST 1:** If the SIO's internal arbiter is enabled through the Arbiter Configuration Register, then this signal is configured as REQ1#. An active low assertion indicates that Initiator1 desires the use of the PCI Bus. If the internal arbiter is disabled, the SIO ignores REQ1# after reset. This pin has a weak internal pull-up resistor.
CPUGNT#	t/s/o	**CPU GRANT:** If the SIO's internal arbiter is enabled, this pin is configured as CPUGNT#. The SIO's internal arbiter asserts CPUGNT# to indicate that the CPU initiator has been granted the PCI Bus. If the internal arbiter is disabled, this signal is meaningless. CPUGNT# is tri-stated from the leading edge of PCIRST#. CPUGNT# is tri-stated until driven by the SIO. CPUGNT# is driven based on the arbitration scheme and the asserted REQx#'s.
GNT0#/ SIOREQ#	t/s/o	GRANT 0/SIO REQUEST: If the SIO's internal arbiter is enabled, this pin is configured as GNT0#. The SIO's internal arbiter asserts GNT0# to indicate that Initiator0 has been granted the PCI Bus. If the internal arbiter is disabled, this pin is configured as SIOREQ#. The SIO asserts SIOREQ# to request the PCI Bus. GNT0#/SIOREQ# is tri-stated from the leading edge of PCIRST#. GNT0#/SIOREQ# is tri-stated until driven by the SIO. GNT0#/SIOREQ# is driven based on the arbitration scheme and the asserted REQx#'s.
GNT1#/ RESUME#	t/s/o	**GRANT 1/RESUME:** If the SIO's internal arbiter is enabled, this pin is configured as GNT1#. The SIO's internal arbiter asserts GNT1# to indicate that Initiator1 has been granted the PCI Bus. If the internal arbiter is disabled, this pin is configured as RESUME#. The SIO asserts RESUME# to indicate that the conditions causing the SIO to retry the cycle has passed. GNT1#/RESUME# is tri-stated from the leading edge of PCIRST#. GNT1#/RESUME# is tri-stated until driven by the SIO. GNT1#/RESUME# is driven based on the arbitration scheme and the asserted REQx#'s.
REQ2#	I	**REQUEST 2:** This pin is an active low signal that indicates that Initiator2 desires the use of the PCI Bus. This signal has a weak internal pull-up resistor.

3.2 PCI Arbiter Signals (Continued)

Signal Name	Type	Description
REQ3#	I	**REQUEST 3:** This pin is an active low signal that indicates that Initiator3 desires the use of the PCI Bus. This signal has a weak internal pull-up resistor.
GNT2#	t/s/o	**GRANT 2:** This pin is configured as GNT2#. The SIO's internal arbiter asserts GNT2# to indicate that Initiator2 has been granted the PCI Bus. GNT2# is high upon reset.
GNT3#	t/s/o	**GRANT 3:** This pin is configured as GNT3#. The SIO's internal arbiter asserts GNT3# to indicate that Initiator3 has been granted the PCI Bus. GNT3# is high upon reset.
MEMREQ#	t/s/o	**MEMORY REQUEST:** If the SIO is configured in Guaranteed Access Time (GAT) Mode, MEMREQ# will be asserted when an ISA master or DMA is requesting the ISA Bus (along with FLSHREQ#) to indicate that the SIO requires ownership of the main memory. MEMREQ# is tri-stated from the leading edge of PCIRST#. MEMREQ# remains tri-stated until driven by the SIO. After PCIRST, MEMREQ# is driven inactive asynchronously from PCIRST# inactive. The SIO asserts FLSHREQ# concurrently with asserting MEMREQ#. **FLSHREQ# / MEMREQ# / Meaning** 1 / 1 / Idle 0 / 1 / Flush buffers pointing towards PCI to avoid ISA deadlock 1 / 0 / Reserved 0 / 0 / GAT mode. Guarantee PCI Bus immediate access to main memory (this may or may not require the PCI-to-main memory buffers to be flushed first depending on the number of buffers).
FLSHREQ#	t/s/o	**FLUSH REQUEST:** FLSHREQ# is generated by the SIO to command all of the system's posted write buffers pointing towards the PCI Bus to be flushed. This is required before granting the ISA Bus to an ISA master or the DMA. FLSHREQ# is tri-stated from the leading edge of PCIRST#. FLSHREQ# remains tri-stated until driven by the SIO. After PCIRST, FLSHREQ# is driven inactive asynchronously from PCIRST# inactive.
MEMACK#	I	**MEMORY ACKNOWLEDGE:** MEMACK# is the response handshake that indicates to the SIO that the function requested over the MEMREQ# and/or FLSHREQ# signals has been completed. In GAT mode (MEMREQ# and FLSHREQ# asserted), the main memory bus is dedicated to the PCI Bus and the system's posted write buffers pointing towards the PCI Bus have been flushed and are disabled. In non-GAT mode (FLSHREQ# asserted alone), this means the system's posted write buffers have been flushed and are disabled. In either case, the SIO can now grant the ISA Bus to the requester.

3.3 Address Decoder Signal

Signal Name	Type	Description
MEMCS#	O	**MEMORY CHIP SELECT:** MEMCS# is a programmable address decode signal provided to a Host CPU bridge. A CPU bridge can use MEMCS# to forward a PCI cycle to main memory behind the bridge. MEMCS# is driven one PCI clock after FRAME# is sampled active (address phase) and is valid for one clock cycle before going inactive. MEMCS# is high upon reset.

3.4 Power Management Signals

Signal Name	Type	Description
SMI#	O	**SYSTEM MANAGEMENT INTERRUPT:** SMI# is an active low output that is asserted by the SIO in response to one of many enableable hardware or software events. SMI# connects directly to the CPU. The SMI# signal is an asychronous input to the CPU. The CPU recognizes the falling edge of SMI# as the highest priority interrupt in the system. The CPU responds by entering SMM (System Management Mode). SMI# is deasserted during and following reset.
STPCLK#	O	**STOP CLOCK**: STPCLK# is an active low output that is asserted by the SIO in response to one of many enableable hardware or software events. STPCLK# connects directly to the CPU. The STPCLK# signal is an asynchronous input to the CPU. When the CPU samples STPCLK# asserted it responds by stopping its internal clock. STPCLK# is deasserted during and following reset.
EXTSMI#	I	**EXTERNAL SYSTEM MANAGEMENT INTERRUPT:** EXTSMI# is a falling edge triggered input to the SIO indicating that an external device is requesting the system to enter SMM mode. When enabled, a falling edge on EXTSMI# will result in the assertion of the SMI# signal to the CPU. EXTSMI# is an asychronous input to the SIO. However, when the setup and hold times are met, it is only required to be asserted for one PCICLK. Once deasserted, it must remain deasserted for at least four PCICLKs in order to allow the edge detect logic to reset. This pin includes a weak internal pull-up resistor.
INIT	I	**INIT:** INIT is an input to the SIO indicating that the CPU is actually being soft reset. It is connected to the INIT pin of the CPU. This pin includes a weak internal pull-up resistor.

3.5 ISA Interface Signals

Signal Name	Type	Description
AEN	O	**ADDRESS ENABLE:** AEN is asserted during DMA cycles to prevent I/O slaves from misinterpreting DMA cycles as valid I/O cycles. When negated, AEN indicates that an I/O slave may respond to address and I/O commands. When asserted, AEN informs I/O resources on the ISA Bus that a DMA transfer is occurring. This signal is also driven high during refresh cycles. AEN is driven low upon reset.

3.5 ISA Interface Signals (Continued)

Signal Name	Type	Description
BALE	O	**BUS ADDRESS LATCH ENABLE:** BALE is an active high signal asserted by the SIO to indicate that the address (SA[19:0], LA[23:17]), AEN and SBHE# signal lines are valid. The LA[23:17] address lines are latched on the trailing edge of BALE. BALE remains asserted throughout DMA and ISA master cycles. BALE is driven low upon reset.
SYSCLK	O	**SYSTEM CLOCK:** SYSCLK is an output of the SIO component. The frequencies supported are 6 MHz to 8.33 MHz.
IOCHRDY	I/O	**I/O CHANNEL READY:** Resources on the ISA Bus assert IOCHRDY to indicate that additional time (wait-states) is required to complete the cycle. This signal is normally high on the ISA Bus. IOCHRDY is an input when the SIO owns the ISA Bus and a PCI agent is accessing an ISA slave or during compatible DMA transfers (compatible cycles only). IOCHRDY is output when an external ISA Bus Master owns the ISA Bus and is accessing a PCI slave or an SIO register. As an SIO output, IOCHRDY is driven inactive (low) from the falling edge of the ISA commands. After data is available for an ISA master read or the SIO latches the data for a write cycle, IOCHRDY is asserted for 70 ns. After 70 ns, the SIO floats IOCHRDY. The 70 ns includes both the drive time and the time it takes the SIO to float IOCHRDY. The SIO does not drive this signal when an ISA Bus master is accessing an ISA Bus slave. IOCHRDY is tri-stated upon reset.
IOCS16#	I	**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.
IOCHK#	I	**I/O CHANNEL CHECK:** IOCHK# can be driven by any resource on the ISA Bus. When asserted, it indicates that a parity or an un-correctable error has occurred for a device or memory on the ISA Bus. A NMI will be generated to the CPU if the NMI generation is enabled.
IOR#	I/O	**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 the SIO owns the ISA Bus. IOR# is an input when an external ISA master owns the ISA Bus. IOR# is driven high upon reset.
IOW#	I/O	**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 the SIO owns the ISA Bus. IOW# is an input when an external ISA master owns the ISA Bus. IOW# is driven high upon reset.
LA[23:17]	I/O	**UNLATCHED ADDRESS:** The LA[23:17] address lines are bi-directional. These address lines allow accesses to physical memory on the ISA Bus up to 16 MBytes. LA[23:17] are outputs when the SIO owns the ISA Bus. The LA[23:17] lines become inputs whenever an ISA master owns the ISA Bus. These signals are undefined during DMA type "A", "B", and "F" cycles. The LA[23:17] signals are at an unknown state upon reset.
SA[19:0]	I/O	**SYSTEM ADDRESS BUS:** These bi-directional address lines define the selection with the granularity of one byte within the one Megabyte section of memory defined by the LA[23:17] address lines. The address lines SA[19:17] that are coincident with LA[19:17] are defined to have the same values as LA[19:17] for all memory cycles. For I/O accesses, only SA[15:0] are used. SA[19:0] are outputs when the SIO owns the ISA Bus. SA[19:0] are inputs when an external ISA Master owns the ISA Bus. SA[19:0] are undefined during DMA type "A", "B", or "F" cycles. SA[19:0] are at an unknown state upon reset.

3.5 ISA Interface Signals (Continued)

Signal Name	Type	Description
SBHE#	I/O	**SYSTEM BYTE HIGH ENABLE:** SBHE# indicates, when asserted, 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 the SIO owns the ISA Bus. SBHE# is an input when an external ISA master owns the ISA Bus. SBHE# is at an unknown state upon reset.
MEMCS16#	OD	**MEMORY CHIP SELECT 16:** MEMCS16# is a decode of LA[23:17] without any qualification of the command signal lines. ISA slaves that are 16-bit memory devices drive this signal low. The SIO ignores MEMCS16# during I/O access cycles and refresh cycles. During DMA cycles, this signal is only used by the byte swap logic. MEMCS16# is an input when the SIO owns the ISA Bus. MEMCS16# is an output when an ISA Bus master owns the ISA Bus. The SIO drives this signal low during ISA master to PCI memory cycles. MEMCS16# is at an unknown state upon reset.
MASTER#	I	**MASTER:** An ISA Bus master asserts MASTER# to indicate that it has control of the ISA Bus. Before the ISA master can assert MASTER#, it must first sample DACK# active. Once MASTER# is asserted, the ISA master has control of the ISA Bus until it negates MASTER#.
MEMR#	I/O	**MEMORY READ:** MEMR# is the command to a memory slave that it may drive data onto the ISA data bus. MEMR# is an output when the SIO is a master on the ISA Bus. MEMR# is an input when an ISA master, other than the SIO, owns the ISA Bus. This signal is also driven by the SIO during refresh cycles. For compatible timing mode DMA cycles, the SIO, as a master, asserts MEMR# if the address is less than 16 MBytes. This signal is not generated for accesses to addresses greater than 16 MByte. MEMR# is not driven active during DMA type "A", "B", or "F" cycles.
MEMW#	I/O	**MEMORY WRITE:** MEMW# is the command to a memory slave that it may latch data from the ISA data bus. MEMW# is an output when the SIO owns the ISA Bus. MEMW# is an input when an ISA master, other than the SIO, owns the ISA Bus. For compatible timing mode DMA cycles, the SIO, as a master, asserts MEMW# if the address is less than 16 MBytes. This signal is not generated for accesses to addresses greater than 16 MByte. MEMW# is not driven active during DMA type "A", "B", or "F" cycles.
SMEMW#	O	**SYSTEM MEMORY WRITE:** The SIO asserts SMEMW# to request a memory slave to accept data from the data lines. If the access is below the 1 MByte range (00000000h–000FFFFFh) during DMA compatible, SIO master, or ISA master cycles, the SIO asserts SMEMW#. SMEMW# is a delayed version of MEMW#. SMEMW# is driven high upon reset.
SMEMR#	O	**SYSTEM MEMORY READ:** The SIO asserts SMEMR# to request a memory slave to accept data from the data lines. If the access is below the 1 MByte range (00000000h–000FFFFFh) during DMA compatible, SIO master, or ISA master cycles, the SIO asserts SMEMR#. SMEMR# is a delay version of MEMR#. Upon PCIRST# this signal is low. SMEMR# is driven high upon reset.

3.5 ISA Interface Signals (Continued)

Signal Name	Type	Description
ZEROWS#	I	**ZERO WAIT-STATES:**An ISA slave asserts ZEROWS# after its address and command signals have been decoded to indicate that the current cycle can be shortened. A 16-bit ISA memory cycle can be reduced to two SYSCLKs. An 8-bit memory or I/O cycle can be reduced to three SYSCLKs. ZEROWS# has no effect during 16-bit I/O cycles. If IOCHRDY and ZEROWS# are both asserted during the same clock, then ZEROWS# is ignored and wait states are added as a function of IOCHRDY (i.e., IOCHRDY has precedence over ZEROWS#).
OSC	I	**OSCILLATOR:** OSC is the 14.31818 MHz ISA clock signal. It is used by the internal 8254 Timer, counters 0, 1, and 2.
RSTDRV	O	**RESET DRIVE:** The SIO asserts RSTDRV to reset devices that reside on the ISA Bus. The SIO asserts this signal when PCIRST# (PCI Reset) is asserted. In addition, the SIO can be programmed to assert RSTDRV by writing to the ISA Clock Divisor Register. Software should assert the RSTDRV during configuration to reset the ISA Bus when changing the clock divisor. Note that when RSTDRV is generated via the ISA Clock Divisor Register, software must ensure that RSTDRV is driven active for a minimum of 1 μs.
SD[15:0]	I/O	**System DATA:** SD[15:0] provide the 16-bit data path for devices residing on the ISA Bus. SD[15:8] correspond to the high order byte and SD[7:0] correspond to the low order byte. SD[15:0] are undefined during refresh. The SIO tri-states SD[15:0] during reset.

3.6 DMA Signals

Signal Name	Type	Description
DREQ [3:0,7:5]	I	**DMA REQUEST:** The DREQ lines are used to request DMA service from the SIO's DMA controller or for a 16-bit master to gain control of the ISA expansion bus. The active level (high or low) is programmed via the DMA Command Register (bit 6). When the bit 6 = 0, DREQ[3:0,7:5] are active high and when bit 6 = 1, the signals are active low. All inactive to active edges of DREQ are assumed to be asynchronous. The request must remain active until the appropriate DACK signal is asserted.
DACK# [3:0,7:5]	O	**DMA ACKNOWLEDGE:** The DACK output lines indicate that a request for DMA service has been granted by the SIO or that a 16-bit master has been granted the bus. The active level (high or low) is programmed via the DMA Command Register (bit 7). When bit 7 = 0, DACK# [3 0,7:5] are active low and when bit 7 = 1, the signals are active high. These lines should be used to decode the DMA slave device with the IOR# or IOW# line to indicate selection. If used to signal acceptance of a bus master request, this signal indicates when it is legal to assert MASTER#. If the DMA controller has been programmed for a timing mode other than compatible mode, and another device has requested the bus, and a 4 μs time has elapsed, this line will be negated and the transfer stopped before the transfer is complete. In this case, the transfer is re-started at the next arbitration period that the channel wins the bus. Upon PCIRST#, these lines are set inactive (high).

3.6 DMA Signals (Continued)

Signal Name	Type	Description
EOP	I/O	**END OF PROCESS:** EOP is bi-directional, acting in one of two modes, and is directly connected to the TC line of the ISA Bus. DMA slaves assert EOP to the SIO to terminate DMA cycles. The SIO asserts EOP to DMA slaves as a terminal count indicator. **EOP-IN MODE:** For all transfer types during DMA, the SIO samples EOP. If it is sampled asserted, the transfer is terminated. **TC-OUT MODE:** The SIO asserts EOP after a new address has been output, if the byte count expires with that transfer. The EOP (TC) remains asserted until AEN is negated, unless AEN is negated during an autoinitialization. EOP (TC) is negated before AEN is negated during an autoinitialization. When all the DMA channels are not in use, the EOP signal is in output mode and negated (low). After PCIRST#, EOP is in output mode and inactive.
REFRESH#	I/O	**REFRESH:** As an output, REFRESH# is used by the SIO to indicate when a refresh cycle is in progress. It should be used to enable the SA[15:0] address to the row address inputs of all banks of dynamic memory on the ISA Bus. Thus, when MEMR# is asserted, the entire expansion bus dynamic memory is refreshed. Memory slaves must not drive any data onto the bus during refresh. As an output, this signal is driven directly onto the ISA Bus. This signal is an output only when the SIO DMA refresh is a master on the bus responding to an internally generated request for refresh. As an input, REFRESH# is driven by 16-bit ISA Bus masters to initiate refresh cycles. Upon PCIRST#, this signal is tri-stated.

3.7 Timer Signal

Signal Name	Type	Description
SPKR	O	**SPEAKER DRIVE:** The SPKR signal is the output of counter 2 and is "ANDed" with Port 061h bit 1 to provide Speaker Data Enable. This signal drives an external speaker driver device, which in turn drives the ISA system speaker. SPKR has a 24 mA drive capability. Upon reset, its output state is 0.

3.8 Interrupt Controller Signals

Signal Name	Type	Description
IRQ[15,14, 11:9, 7:3,1]	I	**INTERRUPT REQUEST:** The IRQ signals provide both system board components and ISA Bus I/O devices with a mechanism for asynchronously interrupting the CPU. The assertion mode of these inputs depends on the programming of LTIM, bit 3 of ICW1 on both Controller-1 and Controller-2. When LTIM is programmed to a 0, a low-to-high transition on any of that controller's IRQ lines is recognized as an interrupt request. This is "edge-triggered" mode. Edge-triggered mode is the SIO default. When LTIM is programmed to a 1, a high level on any of that controller's IRQ lines is recognized as an interrupt request. This mode is "level-triggered" mode. Upon PCIRST#, the IRQ lines are placed in edge-triggered mode. An active IRQ input must remain asserted until after the interrupt is acknowledged. If the input goes inactive before this time, a DEFAULT IRQ7 occurs when the CPU acknowledges the interrupt. **NOTE:** Refer to the Utility Bus Signal descriptions for IRQ12 and IRQ13 signal descriptions.
IRQ8#	I	**INTERRUPT REQUEST EIGHT SIGNAL:** IRQ8# is an active low interrupt input. The assertion mode of this input depends on the programming of the LTIM bit of ICW1 on both Controller-1 and Controller-2. When the LTIM = 0, a high-to-low transition on IRQ8# is recognized as an interrupt request. This is "edge-triggered" mode. Edge triggered mode is the SIO default. When the LTIM = 1, a low level on IRQ8# is recognized as an interrupt request. This mode is "level-triggered" mode. Upon PCIRST#, IRQ8# will be placed in edge-triggered mode. IRQ8# must remain asserted until after the interrupt is acknowledged. If the input goes inactive before this time, a DEFAULT IRQ7 will occur when the CPU acknowledges the interrupt.
INT	O	**CPU INTERRUPT:** INT is driven by the SIO to signal the CPU that an interrupt request is pending and needs to be serviced. It is asynchronous with respect to SYSCLK or PCICLK and is always an output. The interrupt controller must be programmed following a reset to ensure that INT is at a known state. Upon PCIRST#, INT is driven low.
NMI	O	**NON-MASKABLE INTERRUPT:** NMI is used to force a non-maskable interrupt to the CPU. The SIO generates an NMI when either SERR# or IOCHK# is asserted, depending on how the NMI Status and Control Register is programmed. The CPU detects an NMI when it detects a rising edge on NMI. After the NMI interrupt routine processes the interrupt, the NMI status bits in the NMI Status and Control Register are cleared by software. The NMI interrupt routine must read this register to determine the source of the interrupt. The NMI is reset by setting the corresponding NMI source enable/disable bit in the NMI Status and Control Register. To enable NMI interrupts, the two NMI enable/disable bits in the register must be set to 0, and the NMI mask bit in the NMI Enable/Disable and Real-Time Clock Address Register must be set to 0. Upon PCIRST#, this signal is driven low.

3.9 Utility Bus Signals

Signal Name	Type	Description
UBUSTR	O	**UTILITY DATA BUS TRANSMIT/RECEIVE:** UBUSTR is tied directly to the direction control of a 74F245 that buffers the utility data bus, UD[7:0]. UBUSTR is asserted for all I/O read cycles (regardless if a Utility Bus device has been decoded). UBUSTR is asserted for memory cycles only if BIOS space has been decoded. For PCI and ISA master-initiated read cycles, UBUSTR is asserted from the falling edge of either IOR# or MEMR#, depending on the cycle type (driven from MEMR# only if BIOS space has been decoded). When the rising edge of IOR# or MEMR# occurs, the SIO negates UBUSTR. For DMA read cycles from the Utility Bus, UBUSTR is asserted when DACKx# is asserted and negated when DACKx# is negated. At all other times, UBUSTR is negated. Upon PCIRST#, this signal is driven low.
UBUSOE#	O	**UTILITY DATA BUS OUTPUT ENABLE:** UBUSOE# is tied directly to the output enable of a 74F245 that buffers the utility data bus, UD[7:0], from the system data bus, SD[7:0]. UBUSOE# is asserted anytime a SIO supported Utility Bus device is decoded, and the devices decode is enabled in the Utility Bus Chip Select Enable Registers. UBUSOE# is asserted from the falling edge of the ISA commands (IOR#, IOW#, MEMR#, or MEMW#) for PCI and ISA master-initiated cycles. UBUSOE# is negated from the rising edge of the ISA command signals for SIO-initiated cycles and the SA[16:0] and LA[23:17] address for ISA master-initiated cycles. For DMA cycles, UBUSOE# is asserted when DACK2# is asserted and negated when DACK2# negated. UBUSOE# is not driven active under the following conditions: **NOTES:** 1. During an I/O access to the floppy controller, if DSKCHG is sampled low at reset. 2. If the Digital Output Register is programmed to ignore DACK2#. 3. During an I/O read access to floppy location 3F7h (primary) or 377h (secondary), if the IDE decode space is disabled (i.e. IDE is not resident on the Utility Bus). 4. During any access to a utility bus peripheral in which its decode space has been disabled. Upon a PCIRST#, this signal is driven inactive (high).
ECSADDR [2:0]	O	**ENCODED CHIP SELECTS:** ECSADDR[2:0] are the encoded chip selects and/or control signals for the Utility Bus peripherals supported by the SIO. The binary code formed by the three signals indicates which Utility Bus device is selected. These signals tie to the address inputs of two external 74F138 decoder chips and are driven valid/invalid from the SA[16:0] and LA[23:17] address lines. Upon PCIRST#, these signals are driven high.
ECSEN#	O	**ENCODED CHIP SELECT ENABLE:** ECSEN# is used to determine which of the two external 74F138 decoders is to be selected. ECSEN# is driven low to select decoder 1 and driven high to select decoder 2. This signal is driven valid/invalid from the SA[16:0] and LA[23:17] address lines (except for the generation of RTCALE#, in which case, ECSEN# is driven active based on IOW# falling, and remains active for two SYSCLKs). During a non-valid address or during an access not targeted for the Utility Bus, this signal is driven high. Upon PCIRST#, this signal is driven high.

3.9 Utility Bus Signals (Continued)

Signal Name	Type	Description
ALT_RST#	O	**ALTERNATE RESET:** ALT_RST# is used to reset the CPU under program control. This signal is AND'ed together externally with the reset signal (KBDRST#) from the keyboard controller to provide a software means of resetting the CPU. This provides a faster means of reset than is provided by the keyboard controller. Writing a 1 to bit 0 in the Port 92 Register causes this signal to pulse low for approximately 4 SYSCLKs. Before another ALT_RST# pulse can be generated, bit 0 must be set to 0. Upon PCIRST#, this signal is driven inactive high (bit 0 in the Port 92 Register is set to 0).
ALT_A20	O	**ALTERNATE A20:** ALT_A20 is used to force A20M# to the CPU low for support of real mode compatible software. This signal is externally OR'ed with the A20GATE signal from the keyboard controller and CPURST to control the A20M# input of the CPU. Writing a 0 to bit 1 of the Port 92 Register forces ALT_A20 low. ALT_A20 low drives A20M# to the CPU low, if A20GATE from the keyboard controller is also low. Writing a 1 to bit 1 of the Port 92 Register force ALT_A20 high. ALT_A20 high drives A20M# to the CPU high, regardless of the state of A20GATE from the keyboard controller. Upon reset, this signal is driven low.
DSKCHG	I	**DISK CHANGE:** DSKCHG is tied directly to the DSKCHG signal of the floppy controller. This signal is inverted and driven on SD7 during I/O read cycles to floppy address locations 3F7h (primary) or 377h (secondary) as shown in the table below. Note that the primary and secondary locations are programmed in the Utility Bus Address Decode Enable/Disable Register "A".

FLOPPYCS# Decode	IDECSx# Decode	State of SD7 (output)	State of UBUSOE#
Enabled	Enabled	Tri-stated	Enabled
Enabled	Disabled	Driven via DSKCHG	Disabled
Disabled	Enabled	Tri-stated	Enabled (note)
Disabled	Disabled	Tri-stated	Disabled

NOTE:

For this mode to be supported, extra logic is required to disable the U-bus transceiver for accesses to 3F7/377. This is necessary because of potential contention between the Utility Bus buffer and a floppy on the ISA Bus driving the system bus at the same time during shared I/O accesses.

This signal is also used to determine if the floppy controller is present on the Utility Bus. It is sampled on the trailing edge of PCIRST#, and if high, the Floppy is present. For systems that do not support a Floppy via the SIO, this pin should be strapped low. If sampled low, the SD7 function, UBUSOE#, and ECSADDR[2:0] signals will not be enabled for DMA or programmed I/O accesses to the floppy disk controller. This condition overrides the floppy decode enable bits in the Utility Bus Chip Select A.

3.9 Utility Bus Signals (Continued)

Signal Name	Type	Description
FERR#/ IRQ13	I	**NUMERIC COPROCESSOR ERROR/IRQ13:** This signal has two separate functions, depending on bit 5 in the ISA Clock Divisor Register. This pin functions as a FERR# signal supporting coprocessor errors, if this function is enabled (bit 5 = 1), or as an external IRQ13, if the coprocessor error function is disabled (bit 5 = 0). If programmed to support coprocessor error reporting, this signal is tied to the coprocessor error signal on the CPU. If FERR# is asserted by the coprocessor inside the CPU, the SIO generates an internal IRQ13 to its interrupt controller unit. The SIO then asserts the INT output to the CPU. Also, in this mode, FERR# gates the IGNNE# signal to ensure that IGNNE# is not asserted to the CPU unless FERR# is active. When FERR# is asserted, the SIO asserts INT to the CPU as an IRQ13. IRQ13 continues to be asserted until a write to F0h has been detected. If the Coprocessor error reporting is disabled, FERR# can be used by the system as IRQ13. Upon PCIRST#, this signal provides the standard IRQ13 function. This signal should be pulled high with an external 8.2 KΩ pull-up resistor if the IRQ13 mode is used or the pin is left floating.
IGNNE#	O	**IGNORE ERROR:** This signal is connected to the ignore error pin of the CPU. IGNNE# is only used if the SIO coprocessor error reporting function is enabled in the ISA Clock Divisor Register (bit 5 = 1). If FERR# is active, indicating a coprocessor error, a write to the Coprocessor Error Register (F0h) causes the IGNNE# to be asserted. IGNNE# remains asserted until FERR# is negated. If FERR# is not asserted when the Coprocessor Error Register is written, the IGNNE# is not asserted. IGNNE# is driven high upon a reset.
IRQ12/M	I	**INTERRUPT REQUEST/MOUSE INTERRUPT:** In addition to providing the standard interrupt function as described in the pin description for IRQ[15,14, 11:9, 7:3, 1], this pin also provides a mouse interrupt function. Bit 4 in the ISA Clock Divisor Register determines the functionality of IRQ12/M. When bit 4 = 0, the standard interrupt function is provided and this pin can be tied to the ISA connector. When bit 4 = 1, the mouse interrupt function is provided and this pin can be tied to the DIRQ12 output of the keyboard controller. When the mouse interrupt function is selected, a low to high transition on this signal is latched by the SIO and an INT is generated to the CPU as IRQ12. An interrupt will continue to be generated until a PCIRST# or an I/O read access to address 60h (falling edge of IOR#) is detected. After a PCIRST#, this pin provides the standard IRQ12 function.

3.10 Test Signals

Signal Name	Type	Description
TEST	I	**TEST**: The TEST signal is used to tri-state all of the SIO outputs. During normal operation, this input should be tied to ground.
TESTO	O	**TEST OUTPUT**: This is the output pin used during NAND tree testing.

4.0 REGISTER DESCRIPTION

The SIO contains both PCI configuration registers and non-configuration registers. The configuration registers (Table 3) are located in PCI configuration space and are only accessible from the PCI Bus. Addresses for configuration registers are offset values that appear on AD[7:2] and C/BE#[3:0]. The configuration registers (Section 4.1) can be accessed as Byte, Word (16-bit), or Dword (32-bit) quantities. All multi-byte numeric fields use "little-endian" ordering (i.e., lower addresses contain the least significant parts of the fields).

The non-configuration registers (Table 4) include DMA Registers (Section 4.2), Timer Registers (Section 4.3), Interrupt Controller Registers (Section 4.4), and Control Registers (Section 4.5). All of these registers are accessible from the PCI Bus. In addition, some of the registers are accessible from the ISA Bus. Table 4 indicates the bus access for each register. Except for the DMA scatter/gather registers and the BIOS timer registers, the non-configuration registers can only be accessed as byte quantities. If a PCI master attempts a multi-byte access (i.e., more than one Byte Enable signal asserted), the SIO responds with a target-abort. The scatter/gather registers and BIOS timer registers can be accessed as Byte, Word, or Dword quantities.

Some of the SIO configuration and non-configuration registers contain reserved bits. These bits are labeled "Reserved". Software must take care to deal correctly with bit-encoded fields that are reserved. On reads, software must use appropriate masks to extract the defined bits and not rely on reserved bits being any particular value. On writes, software must ensure that the values of reserved bit positions are preserved. That is, the values of reserved bit positions must first be read, merged with the new values for other bit positions, and the data then written back.

In addition to reserved bits within a register, the SIO contains address locations in the PCI configuration space that are marked "Reserved" (Table 3). The SIO responds to accesses to these address locations by completing the PCI cycle. However, reads of reserved address locations yield all zeroes and writes have no affect on the SIO.

The SIO, upon receiving a hard reset (PCIRST# signal), sets its internal registers to pre-determined **default** states. The default values are indicated in the individual register descriptions.

Table 3. Configuration Registers

Configuration Offset	Register	Register Access	Bus Access
00h–01h	Vendor Identification	RO	PCI Only
02h–03h	Device Identification	RO	PCI Only
04h–05h	Command	R/W	PCI Only
06h–07h	Device Status	R/W	PCI Only
08h	Revision Identification	RO	PCI Only
09h–3Fh	Reserved	—	PCI Only
40h	PCI Control	R/W	PCI Only
41h	PCI Arbiter Control	R/W	PCI Only
42h	PCI Arbiter Priority Control	R/W	PCI Only
43h	PCI Arbiter Priority Control Extension Register	R/W	PCI Only
44h	MEMCS# Control	R/W	PCI Only
45h	MEMCS# Bottom of Hole	R/W	PCI Only
46h	MEMCS# Top of Hole	R/W	PCI Only
47h	MEMCS# Top of Memory	R/W	PCI Only
48h	ISA Address Decoder Control	R/W	PCI Only
49h	ISA Address Decoder ROM Block Enable	R/W	PCI Only
4Ah	ISA Address Decoder Bottom of Hole	R/W	PCI Only
4Bh	ISA Address Decoder Top of Hole	R/W	PCI Only
4Ch	ISA Controller Recovery Timer	R/W	PCI Only
4Dh	ISA Clock Divisor	R/W	PCI Only
4Eh	Utility Bus Chip Select Enable A	R/W	PCI Only
4Fh	Utility Bus Chip Select Enable B	R/W	PCI Only
50h–53h	Reserved	—	PCI Only
54h	MEMCS# Attribute Register #1	R/W	PCI Only
55h	MEMCS# Attribute Register #2	R/W	PCI Only
56h	MEMCS# Attribute Register #3	R/W	PCI Only
57h	Scatter/Gather Relocation Base Address	R/W	PCI Only
58h–5Fh	Reserved	—	PCI Only

Table 3. Configuration Registers (Continued)

Configuration Offset	Register	Register Access	Bus Access
60h	PIRQ0# Route Control	R/W	PCI Only
61h	PIRQ1# Route Control	R/W	PCI Only
62h	PIRQ2# Route Control	R/W	PCI Only
63h	PIRQ3# Route Control	R/W	PCI Only
64h–7Fh	Reserved	—	PCI Only
80h–81h	BIOS Timer Base Address	R/W	PCI Only
82h–9Fh	Reserved	—	PCI Only
A0h	SMI Control (SMICNTL)	R/W	PCI Only
A1h	Reserved	—	PCI Only
A2h–A3h	SMI Enable (SMIEN)	R/W	PCI Only
A4h–A7h	System Event Enable (SEE)	R/W	PCI Only
A8h	Fast Off Timer (FTMR)	R/W	PCI Only
A9h	Reserved	—	PCI Only
AAh–ABh	SMI Request (SMIREQ)	R/W	PCI Only
ACh	Clock Throttle STPCLK# Low Timer (CTLTMRL)	R/W	PCI Only
ADh	Reserved	—	PCI Only
AEh	Clock Throttle STPCLK# High Timer (CTLTMRH)	R/W	PCI Only
AFh–FFh	Reserved	—	PCI Only

Table 4. Non-Configuration Registers

Address	Function Unit	Register	Register Access	Bus Access
0000h	DMA	DMA1 CH0 Base and Current Address	R/W	PCI Only
0001h	DMA	DMA1 CH0 Base and Current Count	R/W	PCI Only
0002h	DMA	DMA1 CH1 Base and Current Address	R/W	PCI Only
0003h	DMA	DMA1 CH1 Base and Current Count	R/W	PCI Only
0004h	DMA	DMA1 CH2 Base and Current Address	R/W	PCI Only
0005h	DMA	DMA1 CH2 Base and Current Count	R/W	PCI Only
0006h	DMA	DMA1 CH3 Base and Current Address	R/W	PCI Only
0007h	DMA	DMA1 CH3 Base and Current Count	R/W	PCI Only
0008h	DMA	DMA1 Status(R) Command(W)	R/W	PCI Only
0009h	DMA	DMA1 Write Request	WO	PCI Only
000Ah	DMA	DMA1 Write Single Mask Bit	WO	PCI Only
000Bh	DMA	DMA1 Write Mode	WO	PCI Only
000Ch	DMA	DMA1 Clear Byte Pointer	WO	PCI Only
000Dh	DMA	DMA1 Master Clear	WO	PCI Only
000Eh	DMA	DMA1 Clear Mask	WO	PCI Only
000Fh	DMA	DMA1 Read/Write All Mask Register Bits	R/W	PCI Only
0020h	Interrupt	INT 1 Control	R/W	PCI/ISA
0021h	Interrupt	INT 1 Mask	R/W	PCI/ISA
0040h	Timer	Timer Counter 1–Counter 0 Count	R/W	PCI/ISA
0041h	Timer	Timer Counter 1–Counter 1 Count	R/W	PCI/ISA
0042h	Timer	Timer Counter 1–Counter 2 Count	R/W	PCI/ISA
0043h	Timer	Timer Counter 1 Command Mode	WO	PCI/ISA
0060h(2)	Control	Reset UBus IRQ12	RO	PCI/ISA
0061h	Control	NMI Status and Control	R/W	PCI/ISA
0070h(2)	Control	CMOS RAM Address and NMI Mask	WO	PCI/ISA
0078h-007Bh(3,4,5)	Timer	BIOS Timer	R/W	PCI Only
0080h(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0081h	DMA	DMA Channel 2 Page Register	R/W	PCI/ISA
0082h	DMA	DMA Channel 3 Page Register	R/W	PCI/ISA
0083h	DMA	DMA Channel 1 Page Register	R/W	PCI/ISA

Table 4. Non-Configuration Registers (Continued)

Address	Function Unit	Register	Register Access	Bus Access
0084h(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0085h(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0086h(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0087h	DMA	DMA Channel 0 Page Register	R/W	PCI/ISA
0088h(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0089h	DMA	DMA Channel 6 Page Register	R/W	PCI/ISA
008Ah	DMA	DMA Channel 7 Page Register	R/W	PCI/ISA
008Bh	DMA	DMA Channel 5 Page Register	R/W	PCI/ISA
008Ch(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
008Dh(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
008Eh(1)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
008Fh	DMA	DMA Low Page Register Refresh	R/W	PCI/ISA
0090h(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0092h(2)	Control	Port 92 Register	R/W	PCI/ISA
0094h(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0095h(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0096h(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
0098h(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
009Ch(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
009Dh(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
009Eh(6)	DMA	DMA Page Register Reserved	R/W	PCI/ISA
009Fh	DMA	DMA Low Page Register Refresh	R/W	PCI/ISA
00A0h	Interrupt	INT 2 Control Register	R/W	PCI/ISA
00A1h	Interrupt	INT 2 Mask Register	R/W	PCI/ISA
00B2h	P.M.	Advanced Power Management Control Port	R/W	PCI Only
00B3h	P.M.	Advanced Power Management Status Port	R/W	PCI Only
00C0h	DMA	DMA2 CH0 Base and Current Address	R/W	PCI Only
00C2h	DMA	DMA2 CH0 Base and Current Count	R/W	PCI Only
00C4h	DMA	DMA2 CH1 Base and Current Address	R/W	PCI Only
00C6h	DMA	DMA2 CH1 Base and Current Count	R/W	PCI Only
00C8h	DMA	DMA2 CH2 Base and Current Address	R/W	PCI Only

Table 4. Non-Configuration Registers (Continued)

Address	Function Unit	Register	Register Access	Bus Access
00CAh	DMA	DMA2 CH2 Base and Current Count	R/W	PCI Only
00CCh	DMA	DMA2 CH3 Base and Current Address	R/W	PCI Only
00CEh	DMA	DMA2 CH3 Base and Current Count	R/W	PCI Only
00D0h	DMA	DMA2 Status(r) Command(w) Register	R/W	PCI Only
00D2h	DMA	DMA2 Write Request Register	WO	PCI Only
00D4h	DMA	DMA2 Write Single Mask Bit Register	WO	PCI Only
00D6h	DMA	DMA2 Write Mode Register	WO	PCI Only
00D8h	DMA	DMA2 Clear Byte Pointer Register	WO	PCI Only
00DAh	DMA	DMA2 Master Clear Register	WO	PCI Only
00DCh	DMA	DMA2 Clear Mask Register	WO	PCI Only
00DEh	DMA	DMA2 Read/Write All Mask Register Bits	R/W	PCI Only
00F0h(2)	Control	Coprocessor Error Register	WO	PCI/ISA
0372h(2)	Control	Secondary Floppy Disk Digital Output Register	WO	PCI/ISA
03F2h(2)	Control	Primary Floppy Disk Digital Output Register	WO	PCI/ISA
040Ah(3)	DMA	Scatter/Gather Interrupt Status Register	RO	PCI Only
040Bh	DMA	DMA1 Extended Mode Register	WO	PCI/ISA
0410h(3,4)	DMA	CH0 Scatter/Gather Command	WO	PCI Only
0411h(3,4)	DMA	CH1 Scatter/Gather Command	WO	PCI Only
0412h(3,4)	DMA	CH2 Scatter/Gather Command	WO	PCI Only
0413h(3,4)	DMA	CH3 Scatter/Gather Command	WO	PCI Only
0415h(3,4)	DMA	CH5 Scatter/Gather Command	WO	PCI Only
0416h(3,4)	DMA	CH6 Scatter/Gather Command	WO	PCI Only
0417h(3,4)	DMA	CH7 Scatter/Gather Command	WO	PCI Only
0418h(3,4)	DMA	CH0 Scatter/Gather Status	RO	PCI Only
0419h(3,4)	DMA	CH1 Scatter/Gather Status	RO	PCI Only
041Ah(3,4)	DMA	CH2 Scatter/Gather Status	RO	PCI Only
041Bh(3,4)	DMA	CH3 Scatter/Gather Status	RO	PCI Only
041Dh(3,4)	DMA	CH5 Scatter/Gather Status	RO	PCI Only
041Eh(3,4)	DMA	CH6 Scatter/Gather Status	RO	PCI Only
041Fh(3,4)	DMA	CH7 Scatter/Gather Status	RO	PCI Only
0420h–0423h(3,4)	DMA	CH0 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only

Table 4. Non-Configuration Registers (Continued)

Address	Function Unit	Register	Register Access	Bus Access
0424h–0427h(3,4)	DMA	CH1 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
0428h–042Bh(3,4)	DMA	CH2 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
042Ch–042Fh(3,4)	DMA	CH3 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
0434h–0437h(3,4)	DMA	CH5 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
0438h–043Bh(3,4)	DMA	CH6 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
043Ch–043Fh(3,4)	DMA	CH7 Scatter/Gather Descriptor Table Pointer	R/W	PCI Only
0481h	DMA	DMA CH2 High Page Register	R/W	PCI/ISA
0482h	DMA	DMA CH3 High Page Register	R/W	PCI/ISA
0483h	DMA	DMA CH1 High Page Register	R/W	PCI/ISA
0487h	DMA	DMA CH0 High Page Register	R/W	PCI/ISA
0489h	DMA	DMA CH6 High Page Register	R/W	PCI/ISA
048Ah	DMA	DMA CH7 High Page Register	R/W	PCI/ISA
048Bh	DMA	DMA CH5 High Page Register	R/W	PCI/ISA
04D0h	Interrupt	Edge/Level Control Register—INT CNTRL 1	R/W	PCI Only
04D1h	Interrupt	Edge/Level Control Register—INT CNTRL 2	R/W	PCI Only
04D6h	DMA	DMA2 Extended Mode Register	WO	PCI/ISA

NOTES:
1. PCI write cycles to these address locations flow through to the ISA Bus. PCI read cycles to these address locations do not flow through to the ISA Bus.
2. PCI read and write cycles to these address locations flow through to the ISA Bus.
3. The I/O address of this register is relocatable. The value shown in this table is the default address location.
4. This register can be accessed as a Byte, Word, or Dword quantity.
5. If this register location is enabled, PCI accesses to the BIOS Timer Register do not flow through to the ISA Bus. If disabled, accesses to this address location flow through to the ISA Bus.
6. When the DMAAC bit in the PCI Control Register is '0', the 82378 will alias I/O accesses in the 80h–8Fh range to the 90h–9Fh range. Write accesses to these address locations flow through to the ISA Bus. Read cycles to these address locations do not flow through to the ISA Bus. When DMAAC = 1, the SIO will only respond to the 80h-8Fh range and read and write accesses to these addresses in the 90h–9Fh range will be forwarded from the PCI bus to the ISA Bus (I/O port 92h is always a distinct register in the 90h–9Fh range and is always fully decoded, regardless of the setting of the DMAAC bit).

4.1 SIO Configuration Register Description

This section describes the SIO configuration registers. These registers include the Mandatory Header Registers (located in the first 64 bytes of configuration space) and the SIO specific registers (located from configuration offset 40h–56h).

4.1.1 VID—VENDOR IDENTIFICATION REGISTER

Address Offset: 00h, 01h
Default Value: 8086h
Attribute: Read Only
Size: 16 bits

The VID Register contains the vendor identification number. This 16-bit register combined with the Device Identification Register uniquely identifies any PCI device. Writes to this register have no effect.

Bits[15:0]: Vendor Identification Number
This is a 16-bit value assigned to Intel.

4.1.2 DID—DEVICE IDENTIFICATION REGISTER

Address Offset: 02h, 03h
Default Value: 0484h
Attribute: Read Only
Size: 16 bits

The DID Register contains the device identification number. This register, along with the Vendor ID, uniquely identifies the SIO. Writes to this register have no effect.

Bits[15:0]: Device Identification Number
This is a 16-bit value assigned to the SIO.

4.1.3 COM—COMMAND REGISTER

Address Offset: 04h–05h
Default Value: 0007h
Attribute: Read/Write
Size: 16 bits

Bits[15:5]: Reserved
Read 0.

Bit 4: PMWE (Postable Memory Write Enable)
Enable Postable memory write, memory write and invalidate, and memory read Pre-fetch commands. The SIO does not support these commands as a master or slave so this bit is not implemented. This bit will always be read as a 0.

Bit 3: SCE (Special Cycle Enable)
When this bit is set to a "1", the SIO will recognize PCI Special Cycles. When set to "0", the SIO will ignore all PCI Special Cycles. This bit MUST be enabled in the 82378ZB if the STPCLK feature is being used.

Bit 2: BME (Bus Master Enable)
Since the SIO always requests the PCI Bus on behalf of ISA masters, DMA, or line buffer PCI requests, this bit is hardwired to a 1 and will always be read as a 1.

Bit 1: MSE (Memory Space Enable)
Enables SIO to accept a PCI-originated memory cycle. Since the SIO always responds to PCI-originated memory cycles (and ISA-bound cycles) by asserting DEVSEL#, this bit is hardwired to a 1 and will always be read as a 1.

Bit 0: IOSE (I/O Space Enable)
Enable SIO to accept a PCI-originated I/O cycle. Since the SIO always responds to a master I/O cycle, this bit is hardwired to a 1 and will always be read as a 1.

4.1.4 DS—DEVICE STATUS REGISTER

Address Offset: 06h, 07h
Default Value: 0200h
Attribute: Read/Write
Size: 16 bits

DSR is a 16-bit status register that reports the occurrence of a PCI master-abort by the SIO or a PCI target-abort when the SIO is a master. The register also indicates the SIO DEVSEL# signal timing that is hardwired in the SIO.

Bit 15: Reserved
Read as 0.

Bit 14: SERRS (SERR# Status)
This bit is set by the PCI devices that assert the SERR# signal. Since SERR# is only an input to the SIO, this bit is not implemented and will always be read as 0.

Bit 13: MA (Master-Abort Status)
When the SIO, as a master, generates a master-abort, MA is set to a 1. Software sets MA to 0 by writing a 1 to this bit location.

Bit 12: RTA (Received Target-Abort Status)
When the SIO is a master on the PCI Bus and receives a target-abort, this bit is set to a 1. Software sets RTA to 0 by writing a 1 to this bit location.

Bit 11: STA (Signaled Target-Abort Status)
This bit is set to a 1 by the SIO when it generates a target-abort.

Bits[10:9]: DEVT (SIO DEVSEL# Timing Status)
This 2-bit field defines the timing for DEVSEL# assertion. These read only bits indicate the SIO's DEVSEL# timing when performing a positive decode. Since the SIO always generates DEVSEL# with medium timing, DEVT = 01. This DEVSEL# timing does not include Configuration cycles.

Bits[8:0]: Reserved
Read as 0's.

4.1.5 RID—REVISION IDENTIFICATION REGISTER

Address Offset: 08h
Default Value: xxh (dependent on Part Revision)
Attribute: Read Only
Size: 4 bits (upper nibble reserved)

This 4-bit register contains the revision number for the SIO. This number indicates the stepping number of the component. Additionally, the upper nibble of the value is reserved. BIOS should mask the upper nibble when reading this register. These bits are read only. Writes to this register have no effect.

Bits[7:4]: Reserved
These 4 bits are reserved.

Bits[3:0]: Revision Identification Number
This is an 4-bit value that indicates the revision identification number for the SIO. Numbers used so far include:
0h: 82378IB A0-Stepping
1h: 82378IB B0-Stepping
WAS NOT IMPLEMENTED. B0 steppings read 0h also. Read the BIOS Timer Base Address Configuration Register to identify between A0 and B0 steppings.
A0 = 0000h
B0 = 0078h
3h: 82378ZB A0-Stepping

4.1.6 PCICON—PCI CONTROL REGISTER

Address Offset: 40h
Default Value: 20h
Attribute: Read/Write
Size: 8 bits

This 8-bit register controls the Line Buffer operation, the SIO's PCI Posted Write Buffer enabling, and the DEVSEL# signal sampling point. The PCICON Register also controls how the SIO responds to INTA cycles on the PCI Bus and if the reserved DMA page registers are aliased from 80h–8Fh to 90h–9Fh.

Bit 7: Reserved
Read as 0.

Bit 6: DMAAC (DMA Reserved Page Register Aliasing Control)
These register bits control whether the SIO will alias I/O accesses in the 80h–8Fh to the 90h–9Fh range. When DMAAC = 0, the SIO will alias I/O accesses in the 80h–8Fh to the 90h–9Fh range (AD4 is not used for decoding the DMA reserved page registers). When DMAAC = 1, the SIO will only respond to the 80h–8Fh range (AD4 is used for decoding the DMA reserved page registers). Read and write accesses to the 90h–9Fh range will be forwarded from the PCI bus to the ISA bus.

NOTE:
I/O port 92h is always a distinct register in the 90h–9Fh range and is always fully decoded, regardless of the setting of this bit.

Bit 5: IAE (Interrupt Acknowledge Enable)
When IAE = 0, the SIO ignores INTA cycles generated on the PCI Bus. However, when disabled, the SIO still responds to accesses to the 8259's register set and allows poll mode functions. When IAE = 1, the SIO responds to INTA cycles in the normal fashion. This bit defaults to a 1 (respond to INTA cycles).

Bits[4:3]: SDSP (Subtractive Decoding Sample Point)
The SDSP field determines the DEVSEL# sample point, after which an inactive DEVSEL# results in the SIO forwarding the unclaimed PCI cycle to the ISA Bus (subtractive decoding). This setting should match the slowest device in the system.

Bit	4	3	Operation
	0	0	Slow sample point
	0	1	Typical sample point
	1	0	Fast sample point
	1	1	Reserved

Bit 2: PPBE (PCI Posted Write Buffer Enable)
When PPBE = 0, the PCI posted write buffer is disabled. When PPBE = 1, the PCI posted write buffer is enabled This bit defaults to disabled mode (PPBE = 0).

Bit 1: ILBC (ISA Master Line Buffer Configuration)
When ILBC = 0, the Line Buffer is in single transaction mode. When ILBC = 1, the Line Buffer is in 8-byte mode. This bit applies only to ISA Master transfers. This bit defaults to single transaction mode (ILBC = 0).

Bit 0: DLBC (DMA Line Buffer Configuration)
When DLBC = 0, the Line Buffer is in single transaction mode. When DLBC = 1, the Line Buffer is in 8-byte mode. This bit applies only to DMA transfers. This bit defaults to single transaction mode (DLBC = 0).

4.1.7 PAC—PCI ARBITER CONTROL REGISTER

Address Offset: 41h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

This 8-bit register controls the operation of the PC arbiter. The PAC register enables/disables the guaranteed access time mode, controls bus lock cycles, enables/disables CPU bus parking, and controls the master retry timer.

Bits[7:5]: Reserved
Read as 0's.

Bits[4:3]: MRT (Master Retry Timer)
This 2-bit field determines the number of PCICLKs after the first retry that a PCI initiator's Bus request will be unmasked.

Bit	4	3	Operation
	0	0	Timer disabled, retries never masked.
	0	1	Retries unmasked after 16 PCICLK's.
	1	0	Retries unmasked after 32 PCICLK's.
	1	1	Retries unmasked after 64 PCICLK's.

Bit 2: BP (Bus Park)
Set to a 1 the SIO will park CPUREQ# on the PCI bus when it detects the PCI bus idle. If Bus Park is disabled, the SIO takes responsibility for driving AD, C/BE# and PAR upon detection of bus idle state if the internal arbiter is enabled.

Bit 1: BL (Bus Lock)
This bit selects between bus lock and resource lock. When BL = 1, Bus Lock is selected. The arbiter considers the entire PCI bus locked upon initiation of any locked transaction. When BL = 0, resource lock is enabled. A locked agent is considered a locked resource and other agents may continue normal PCI transactions.

Bit 0: GAT (Guaranteed Access Time)
This bit enables/disables the guaranteed access time mode. When GAT = 1, the SIO is configured for Guaranteed Access Time mode. This mode is available in order to guarantee the 2.5 μs CHRDY time-out specification for the ISA Bus. When the SIO is an Initiator on behalf of an ISA master, the PCI and memory busses are arbitrated for in serial and must be owned before the ISA master is given ownership of the ISA Bus. When GAT = 0, the guaranteed access time mode is disabled. When guaranteed access time mode is disabled, the ISA master is first granted the ISA Bus and then the SIO arbitrates for the PCI Bus.

4.1.8 PAPC—PCI ARBITER PRIORITY CONTROL REGISTER

Address Offset: 42h
Default Value: 04h
Attribute: Read/Write
Size: 8 bits

This register controls the PCI arbiter priority scheme. The arbiter supports six masters arranged through four switching banks. This permits the six masters to be arranged in a purely rotating priority scheme, one of twenty-four fixed priority schemes, or a hybrid combination of the fixed and rotating priority schemes. Bits[4:7] enable/disable rotate priority for the four banks. For each bit, a 1 enables the mode and a 0 disables the mode. If both fixed and rotate modes are enabled for the same bank, the bank will be in rotate mode. For example, if both bits 0 and 4 are set to a 1, bank 0 will be in rotate mode.

Bit 7: Bank 3 Rotate Control

Bit 6: Bank 2 Rotate Control

Bit 5: Bank 1 Rotate Control

Bit 4: Bank 0 Rotate Control

Bit 3: Bank 2 Fixed Priority Mode Select B

Bit 2: Bank 2 Fixed Priority Mode Select A

Bit 1: Bank 1 Fixed Priority Mode Select

Bit 0: Bank 0 Fixed Priority Mode Select

Fixed Priority Mode

The fixed bank control bits select which requester is the highest priority device within that particular bank.

Table 5. Fixed Mode Bank Control Bits

Mode	Bank					Priority					
	3	2b	2a	1	0	Highest					Lowest
00	0	0	0	0	0	SIOREQ#	REQ0#	REQ2#	REQ3#	CPUREQ#	REQ1#
01	0	0	0	0	1	REQ0#	SIOREQ#	REQ2#	REQ3#	CPUREQ#	REQ1#
02	0	0	0	1	0	SIOREQ#	REQ0#	REQ2#	REQ3#	REQ1#	CPUREQ#
03	0	0	0	1	1	REQ0#	SIOREQ#	REQ2#	REQ3#	REQ1#	CPUREQ#
04	0	0	1	0	0	CPUREQ#	REQ1#	SIOREQ#	REQ0#	REQ2#	REQ3#
05	0	0	1	0	1	CPUREQ#	REQ1#	REQ0#	SIOREQ#	REQ2#	REQ3#
06	0	0	1	1	0	REQ1#	CPUREQ#	SIOREQ#	REQ0#	REQ2#	REQ3#
07	0	0	1	1	1	REQ1#	CPUREQ#	REQ0#	SIOREQ#	REQ2#	REQ3#
08	0	1	0	0	0	REQ2#	REQ3#	CPUREQ#	REQ1#	SIOREQ#	REQ0#
09	0	1	0	0	1	REQ2#	REQ3#	CPUREQ#	REQ1#	REQ0#	SIOREQ#
0A	0	1	0	1	0	REQ2#	REQ3#	REQ1#	CPUREQ#	SIOREQ#	REQ0#
0B	0	1	0	1	1	REQ2#	REQ3#	REQ1#	CPUREQ#	REQ0#	SIOREQ#
0C–0F	0	1	1	x	x	Reserved					
10	1	0	0	0	0	SIOREQ#	REQ0#	REQ3#	REQ2#	CPUREQ#	REQ1#
11	1	0	0	0	1	REQ0#	SIOREQ#	REQ3#	REQ2#	CPUREQ#	REQ1#
12	1	0	0	1	0	SIOREQ#	REQ0#	REQ3#	REQ2#	REQ1#	CPUREQ#
13	1	0	0	1	1	REQ0#	SIOREQ#	REQ3#	REQ2#	REQ1#	CPUREQ#
14	1	0	1	0	0	CPUREQ#	REQ1#	SIOREQ#	REQ0#	REQ3#	REQ2#
15	1	0	1	0	1	CPUREQ#	REQ1#	REQ0#	SIOREQ#	REQ3#	REQ2#
16	1	0	1	1	0	REQ1#	CPUREQ#	SIOREQ#	REQ0#	REQ3#	REQ2#
17	1	0	1	1	1	REQ1#	CPUREQ#	REQ0#	SIOREQ#	REQ3#	REQ2#
18	1	1	0	0	0	REQ3#	REQ2#	CPUREQ#	REQ1#	SIOREQ#	REQ0#
19	1	1	0	0	1	REQ3#	REQ2#	CPUREQ#	REQ1#	REQ0#	SIOREQ#
1A	1	1	0	1	0	REQ3#	REQ2#	REQ1#	CPUREQ#	SIOREQ#	REQ0#
1B		1	0	1	1	REQ3#	REQ2#	REQ1#	CPUREQ#	REQ0#	SIOREQ#
1C–1F	1	1	1	x	x	Reserved					

Rotating Priority Mode
When any Bank Rotate Control bit is set to a one, that particular bank rotates between the two requesting inputs. Any or all banks can be set in rotate mode. If, within a rotating bank, the highest priority input does not have an active request, then the lower priority input will be granted the bus. However, this does not change the rotation scheme. When the bank toggles, the previously lowest priority input will become the highest priority input. Because of this, the maximum latency a device may encounter would be two complete rotations.

4.1.9 ARBPRIX—PCI ARBITER PRIORITY CONTROL EXTENSION REGISTER

Address Offset: 43h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

This register provides the Fixed Priority Mode select for Bank 3 of the arbiter. The ARBPRIX Register fields are shown.

Bits[7:1]: Reserved
Read as 0.

Bit 0: Bank 3 Fixed Priority Mode Select

0 = REQ2# higher priority
1 = REQ3# higher priority

4.1.10 MEMCSCON—MEMCS# CONTROL REGISTER

Address Offset: 44h
Default value: 00h
Attribute: Read/Write
Size: 8 bits

Bits 0-2 of this register enable MEMCS# blocks. PCI addresses within the enabled blocks result in the generation of MEMCS#. Note that the 0–512 KByte segment does not have RE and WE bits. The 0–512 KByte segment can only be turned off with the MEMCS# Master Enable bit (bit 4). Note also, that when the RE and WE bits are both 0 for a particular segment, the PCI master can not access the segment.

Bits[7:5]: Reserved
Read as 0's.

Bit 4: MEMCS# Master Enable
When the MEMCS# master enable bit is set to a 1, the SIO asserts MEMCS# for all accesses to the defined MEMCS# region (that have been programmed in this register and the MAR1, MAR2, and MAR3 Registers). Also, when this bit is a 1, the positive decoding functions enabled by having the ISA Clock Divisor Register bit 6 = 1 and the Utility Bus Chip Select Register "A" bit 6 = 1 are ignored. Subtractive decoding is provided for these memory areas, instead. When the MEMCS# master enable bit is set to a 0, the entire MEMCS# function is disabled. When this bit is 0, MEMCS# will never be asserted.

Bit 3: Write Enable For 0F0000h–0FFFFFh (Upper 64 KByte BIOS)
When this bit is set to a 1, the SIO generates MEMCS# for PCI master memory write accesses to the address range 0F0000h–0FFFFFh. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory write accesses to the address range 0F0000h–0FFFFFh.

Bit 2: Read Enable For 0F0000h–0FFFFFh (Upper 64 KByte BIOS)
When this bit is set to a 1, the SIO generates MEMCS# for PCI master memory read accesses to the address range 0F0000h–0FFFFFh. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory read accesses to the address range 0F0000h–0FFFFFh.

Bit 1: Write Enable For 080000h–09FFFFh (512 KByte–640 KByte)
When this bit is set to a 1, the SIO generates MEMCS# for PCI master memory write accesses to the address range 080000h–09FFFFh. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory write accesses to the address range 080000h–09FFFFh.

Bit 0: Read Enable For 080000h–09FFFFh (512 KByte–640 KByte)
When this bit is set to a 1, the SIO generates MEMCS# for PCI master memory read accesses to the address range 080000h–09FFFFh. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory read accesses to the address range 080000h–09FFFFh.

4.1.11 MEMCSBOH—MEMCS# BOTTOM OF HOLE REGISTER

Address Offset: 45h
Default value: 10h
Attribute: Read/Write
Size: 8 bits

This register defines the bottom of the MEMCS# hole. MEMCS# is not generated for accesses to addresses within the hole defined by this register and the MCSTOH Register. The hole is defined by the following equation: TOH ≥ address ≥ BOH. TOH is the top of the MEMCS# hole defined by the MCSTOH Register and BOH is the bottom of the MEMCS# hole defined by this register.

For example, to program the BOH at 1 MByte, the value of 10h should be written to this register. To program the BOH at 2 MByte + 64 KByte this register should be programmed to 21h. To program the BOH at 8 MByte this register should be programmed to 80h.

When the TOH < BOH the hole is effectively disabled. It is the responsibility of the programmer to guarantee that the BOH is at or above 1 MB. AD[31:24] must be 0's for the hole, meaning the hole is restricted to be under the 16 MByte boundary. The default value for the BOH and TOH effectively disables the hole.

Bit 7: AD23

Bit 6: AD22

Bit 5: AD21

Bit 4: AD20

Bit 3: AD19

Bit 2: AD18

Bit 1: AD17

Bit 0: AD16

4.1.12 MEMCSTOH—MEMCS# TOP OF HOLE REGISTER

Address Offset: 46h
Default value: 0Fh
Attribute: Read/Write
Size: 8 bits

This register defines the top of the MEMCS# hole. MEMCS# is not generated for accesses to addresses within the hole defined by this register and the MCSBOH Register. The hole is defined by the following equation: TOH ≥ address ≥ BOH. TOH is the top of the MEMCS# hole defined by this register and BOH is the bottom of the MEMCS# hole defined by the MCSBOH Register.

For example, to program the TOH at 1 MByte + 64 KByte, this register should be programmed to 10h. To program the TOH at 2 MByte + 128 KByte this register should be programmed to 21h. To program the TOH at 12 MByte this register should be programmed to BFh.

When the TOH < BOH the hole is effectively disabled. It is the responsibility of the programmer to guarantee that the TOH is above 1 MByte. AD[31:24] must be 0's for the hole, meaning the hole is restricted to be under the 16 MByte boundary. The default value for the BOH and TOH effectively disables the hole.

Bit 7: AD23

Bit 6: AD22

Bit 5: AD21

Bit 4: AD20

Bit 3: AD19

Bit 2: AD18

Bit 1: AD17

Bit 0: AD16

4.1.13 MEMCSTOM—MEMCS# TOP OF MEMORY REGISTER

Address Offset: 47h
Default value: 00h
Attribute: Read/Write
Size: 8 bits

This register determines MEMCS# top of memory boundary. The top of memory boundary ranges up to 512 MBytes, in 2 MByte increments. This register is typically set to the top of main memory. Accesses ≥ 2 MByte and ≤ top of memory boundary results in the assertion of the MEMCS# signal (unless the address resides in the hole programmed by the MCSBOH and MCSTOH Registers). A value of 00h disables this 2 MByte-to-top of memory region. A value of 00h assigns the top of memory to include 2 MByte − 1. A value of FFh assigns the top of memory to include 512 MByte − 1.

Bit 7: AD28

Bit 6: AD27

Bit 5: AD26

Bit 4: AD25

Bit 3: AD24

Bit 2: AD23

Bit 1: AD22

Bit 0: AD21

4.1.14 IADCON—ISA ADDRESS DECODER CONTROL REGISTER

Address Offset: 48h
Default value: 01h
Attribute: Read/Write
Size: 8 bits

This register enables the forwarding of ISA or DMA memory cycles to the PCI Bus. In addition, this register sets the top of the "1 MByte to top of main memory" region.

Bits[7:4]:
The top can be assigned in 1 MByte increments from 1 MByte up to 16 MByte. ISA master or DMA accesses within this region are forwarded to PCI unless they are within the hole.

Bits	7	6	5	4	Top of Memory
	0	0	0	0	1 MByte
	0	0	0	1	2 MByte
	0	0	1	0	3 MByte
	0	0	1	1	4 MByte
	0	1	0	0	5 MByte
	0	1	0	1	6 MByte
	0	1	1	0	7 MByte
	0	1	1	1	8 MByte
	1	0	0	0	9 MByte
	1	0	0	1	10 MByte
	1	0	1	0	11 MByte
	1	0	1	1	12 MByte
	1	1	0	0	13 MByte
	1	1	0	1	14 MByte
	1	1	1	0	15 MByte
	1	1	1	1	16 MByte

Bits[3:0]:
ISA and DMA Memory Cycle To PCI Bus Enables. The memory block is enabled by writing a 1 to the corresponding bit position. Setting the bit to 0 disables the corresponding block. ISA or DMA memory cycles to the enabled blocks result in the ISA cycle being forwarded to the PCI Bus. The BIOSCS# enable bit (bit 6 in the UBCSA Register) for the 896K–960K region overrides the function of bit 3 of this register. If the BIOSCS# bit is set to a 1, the ISA or DMA memory cycle is always contained to ISA, regardless of the setting of bit 3 in this register. If the BIOSCS# bit is disabled, the cycle is forwarded to the PCI bus if bit 3 in this register is enabled. Refer to Section 5.5.1.2 for a complete description of BIOS decoding.

Bit	Memory Block
0	0–512 KByte Memory
1	512–640 KByte Memory
2	640–768 KByte VGA Memory
3	896–960 KByte Low BIOS

4.1.15 IADRBE—ISA ADDRESS DECODER ROM BLOCK ENABLE REGISTER

Address Offset: 49h
Default value: 00h
Attribute: Read/Write
Size: 8 bits

ISA addresses within the enabled ranges result in the ISA memory cycle being forwarded to the PCI Bus. For each bit position, the memory block is enabled if the bit is set to 1 and is disabled if the bit is set to 0. If the memory block is disabled, the ISA cycle is not forwarded to the PCI Bus.

Bit 7: 880–896K Memory Enable

Bit 6: 864–880K Memory Enable

Bit 5: 848–864K Memory Enable

Bit 4: 832–848K Memory Enable

Bit 3: 816–832K Memory Enable

Bit 2: 800–816K Memory Enable

Bit 1: 784–800K Memory Enable

Bit 0: 768–784K Memory Enable

4.1.16 IADBOH—ISA ADDRESS DECODER BOTTOM OF HOLE REGISTER

Address Offset: 4Ah
Default value: 10h
Attribute: Read/Write
Size: 8 bits

This register defines the bottom of the ISA Address Decoder hole. The hole is defined by the following equation: TOH $\geq$ address $\geq$ BOH, where BOH is the bottom of the hole address programmed into this register and TOH is the top of the hole address programmed into the IADTOH Register. ISA master or DMA addresses falling within the hole will not be forwarded to the PCI Bus. The hole can be sized in 64 KByte increments and placed anywhere between 1 MByte and 16 MByte on any 64 KByte boundary. It is the responsibility of the programmer to guarantee that the BOH is at or above 1 MByte. A[31:24] must be 0's for the hole, meaning the hole is restricted to be under the 16 MByte boundary. When TOH $<$ BOH, the hole is effectively disabled. The default value for the BOH and TOH disables the hole.

For example, to program the BOH at 1 MByte, this register should be set to 10h. To program the BOH at 2 MBytes, this register should be set to 20h. To program the BOH at 8 MBytes, this register should be set to 80h. These settings are shown in Figure 2.

Bit 7: A23

Bit 6: A22

Bit 5: A21

Bit 4: A20

Bit 3: A19

Bit 2: A18

Bit 1: A17

Bit 0: A16

4.1.17 IADTOH—ISA ADDRESS DECODER TOP OF HOLE REGISTER

Address Offset: 4Bh
Default value: 0Fh
Attribute: Read/Write
Size: 8 bits

This register defines the top of the ISA Address Decoder hole. The hole is defined by the following equation: TOH ≥ address ≥ BOH, where BOH is the bottom of the hole address programmed into the LADBOH Register and TOH is the top of the hole address programmed into this Register. ISA master or DMA addresses falling within the hole will not be forwarded to the PCI Bus. The hole can be sized in 64 KByte increments and placed anywhere between 1 MByte and 16 MByte on any 64 KByte boundary. It is the responsibility of the programmer to guarantee that the TOH is at or above 1 MByte. A[31:24] must be 0's for the hole, meaning the hole is restricted to be under the 16 MByte boundary. When TOH < BOH, the hole is disabled. The default value for the BOH and TOH disables the hole.

For example, to program the TOH at 1 MByte + 64 KByte, this register should be set to 10h. To program the TOH at 2 MByte + 128 KByte, this register should be set to 21h. To program the TOH at 12 MByte, this register should be set to BFh. These settings are shown in Figure 2.

Bit 7: A23

Bit 6: A22

Bit 5: A21

Bit 4: A20

Bit 3: A19

Bit 2: A18

Bit 1: A17

Bit 0: A16

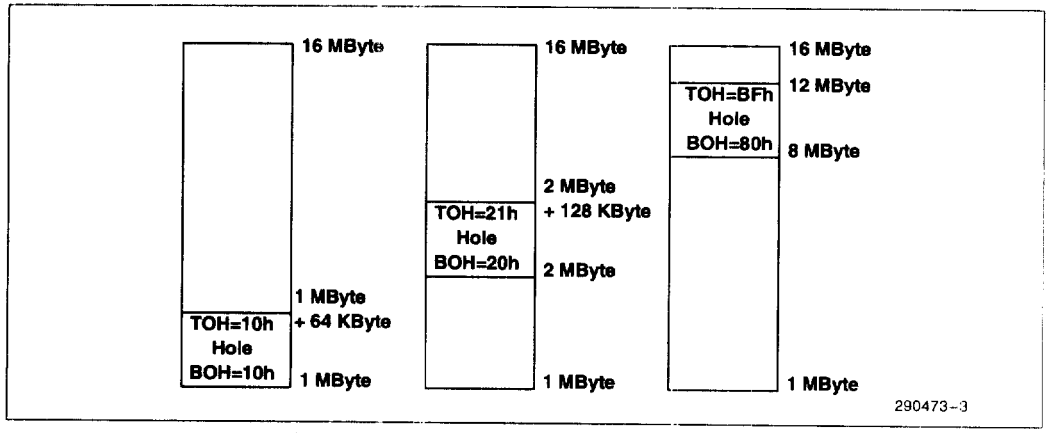


Figure 2. ISA Address Decoder Hole Examples

Table 6. Examples of ISA Decoding

Test Case Description	TOM (48h)	TOH (4Bh)	BOH (4Ah)	Address (hex)	Address	Result
8MB TOM, no hole @ 1M	7xh	0Fh	10h	01000000h 00FFFFFFh 00800000h 007FFFFFh 00100000h 000FFFFFh	16MB 16MB-1 8MB 8MB-1 1MB 1MB-1	To PCI ISA ISA To PCI To PCI ISA (BIOS)
4MB TOM, no hole @ 2M	3xh	1Fh	20h	01000000h 00FFFFFFh 00400000h 003FFFFFh 00200000h 001FFFFFh 00100000h	16MB 16MB-1 4MB 4MB-1 2MB 2MB-1 1MB	To PCI ISA ISA To PCI To PCI To PCI To PCI
1MB TOM, no hole @ 1M	0xh	0Fh	10h	01000000h 00FFFFFFh 00100000h 000FFFFFh	16MB 16MB-1 1MB 1MB-1	To PCI ISA ISA ISA (BIOS)
16MB TOM, 64KB hole @ 15MB	Fxh	F0h	F0h	01000000h 00FFFFFFh 00F10000h 00F0FFFFh 00F00000h 00EFFFFFh 00E10000h 00E0FFFFh 00E00000h 00DFFFFFh	16MB 16MB-1 15MB + 64KB 15MB + 64KB-1 15MB 15MB-1 14MB + 64KB 14MB + 64KB-1 14MB 14MB-1	To PCI To PCI To PCI ISA ISA To PCI To PCI To PCI To PCI To PCI
12MB TOM, 2MB + 128KB hole @ 2MB	Bxh	21h	20h	01000000h 00FFFFFFh 00C00000h 00BFFFFFh 00220000h 0021FFFFh 00210000h 0020FFFFh 00200000h 001FFFFFh 00100000h	16MB 16MB-1 12MB 12MB-1 2MB + 128KB 2MB + 128KB-1 2MB + 64KB 2MB + 64KB-1 2MB 2MB-1 1MB	To PCI ISA ISA To PCI To PCI ISA ISA ISA ISA To PCI To PCI

Table 6. Examples of ISA Decoding (Continued)

Test Case Description	TOM (46h)	TOH (4Bh)	BOH (4Ah)	Address (hex)	Address	Result
5MB TOM, 3MB hole @	4xh	47h	18h	01000000h	16MB	To PCI
				00FFFFFFh	16MB-1	ISA
				00500000h	5MB	ISA
				004FFFFFh	5MB-1	To PCI
				00480000h	4.5MB	To PCI
				0047FFFFh	4.5MB-1	ISA
				00180000h	1.5MB	ISA
				0017FFFFh	1.5MB-1	To PCI
				00100000h	1MB	To PCI

1

4.1.18 ICRT—ISA CONTROLLER RECOVERY TIMER REGISTER

Address Offset: 4Ch
Default Value: 56h
Attribute: Read/Write
Size: 8 bits

The I/O recovery mechanism in the SIO is used to add additional recovery delay between PCI originated 8-bit and 16-bit I/O cycles to the ISA bus. The SIO automatically forces a minimum delay of five SYSCLKs between back-to-back 8- and 16-bit I/O cycles to the ISA bus. The delay is measured from the rising edge of the I/O command (IOR# or IOW#) to the falling edge of the next BALE. If a delay of greater than five SYSCLKs is required, the ISA I/O Recovery Time Register can be programmed to increase the delay in increments of SYSCLKs. Note that no additional delay is inserted for back-to-back I/O "sub cycles" generated as a result of byte assembly or disassembly. This register defaults to 8- and 16-bit recovery enabled with two clocks added to the standard I/O recovery.

Bit 7: Reserved
Read as 0.

Bit 6: 8-Bit I/O Recovery Enable
This bit enables the recovery times programmed into bits 0 and 1 of this register. When this bit is set to 1, the recovery times shown for bits[5:3] are enabled. When this bit is set to 0, recovery times are disabled.

Bits[5:3]: 8-Bit I/O Recovery Times
This 3-bit field defines the recovery times for 8-bit I/O. Programmable delays between back-to-back 8-bit PCI cycles to ISA I/O slaves is shown in terms of ISA clock cycles (SYSCLK) added to the five minimum. The selected delay programmed into this field is enabled/disabled via bit 6 of this register.

Bit	5	4	3	SYSCLK Added	Total SYSCLKs
	0	0	1	+1	6
	0	1	0	+2	7
	0	1	1	+3	8
	1	0	0	+4	9
	1	0	1	+5	10
	1	1	0	+6	11
	1	1	1	+7	12
	0	0	0	+8	13

Bit 2: 16-Bit I/O Recovery Enable
This bit enables the recovery times programmed into bits 0 and 1 of this register. When this bit is set to 1, the recovery times shown for bits 0 and 1 are enabled. When this bit is set to 0, recovery times are disabled.

Bits[1:0]: 16-Bit I/O Recovery Times
This 2-bit field defines the recovery time for 16-bit I/O. Programmable delays between back-to-back 16-bit PCI cycles to ISA I/O slaves is shown in terms of ISA clock cycles (SYSCLK) added to the five minimum. The selected delay programmed into this field is enabled/disabled via bit 2 of this register.

Bit	1	0	SYSCLK Added	Total SYSCLKs
	0	1	+1	6
	1	0	+2	7
	1	1	+3	8
	0	0	+4	9

4.1.19 ICD—ISA CLOCK DIVISOR REGISTER

Address Offset: 4Dh
Default Value: 40h
Attribute: Read/Write
Size: 8 bits

This register selects the integer value used to divide the PCI clock (PCICLK) to generate the ISA clock (SYSCLK). In addition, this register provides an ISA Reset bit to software control RSTDRV, a bit to enable/disable the MOUSE function, a bit to enable/disable the coprocessor error support, and a bit to disable the positive decode for the upper 64 KBytes of BIOS at the top of 1 MByte (F0000h–FFFFFh) and aliased regions.

Bit 7: Reserved

Bit 6: Positive Decode of Upper 64 KByte BIOS Enable
This bit enables (bit 6 = 1) and disables (bit 6 = 0) the positive decode of the upper 64 KBytes of BIOS area at the top of 1 MByte (F0000h–FFFFFh) and the aliased regions at the top of 4 GBytes (FFFF0000h–FFFFFFFFh) and 4 GByte-1 MByte (FFEF0000–FFEFFFFFh). When bit 6 = 1, these address regions are positively decoded, unless bit 4 in the MEMCS# Control Register is set to a 1 in which case these regions are subtractively decoded. When bit 6 = 0, these address regions are subtractively decoded. The encoded chip selects for BIOSCS# and the UBUSOE# signal will always be generated when these locations are accessed, regardless of the state of this bit. A reset sets this bit to a 1 (positive decode enabled).

Bit 5: Coprocessor Error Enable
This bit is used to enable and disable the Coprocessor error support. When enabled (bit 5 = 1), the FERR# input, when driven active, triggers an IRQ13 to the SIO's interrupt controller. FERR# is also used to gate the IGNNE# output. When disabled (bit 5 = 0), the FERR# signal can be used as IRQ13 and the coprocessor support is disabled. A reset sets this bit to 0 (coprocessor support disabled).

Bit 4: IRQ12/M Mouse Function Enable
When this bit is set to 1, IRQ12/M provides the mouse function. When this bit is set to 0, IRQ12/M provides the standard IRQ12 interrupt function. A hard reset sets this bit to 0.

Bit 3: RSTDRV Enable
This bit is used to enable RSTDRV on the ISA Bus. When this bit is set to 1, RSTDRV is asserted and remains asserted until this bit is set to a 0. When set to 0, normal operation of RSTDRV is provided. This bit should be used during configuration to reset the ISA Bus when changing the clock divisor. For a reset, this bit defaults to 0. Note that the software must ensure that RSTDRV is asserted for a minimum of 1 μs.

Bit[2:0]: PCICLK-to-ISA SYSCLK Divisor
These bits are used to select the integer that is used to divide the PCICLK down to generate the ISA SYSCLK. Upon reset, these bits are set to 000 (divisor of 4 selected). For PCI frequencies less than 33 MHz (not including 25 MHz), a clock divisor value must be selected that ensures that the ISA Bus frequency does not violate the 6 MHz to 8.33 MHz SYSCLK specification.

Bit	2	1	0	Divisor	SYSCLK
	0	0	0	4 (33 MHz)	8.33 MHz
	0	0	1	3 (25 MHz)	8.33 MHz
	0	1	0	Reserved	
	0	1	1	Reserved	
	1	0	0	Reserved	
	1	0	1	Reserved	
	1	1	0	Reserved	
	1	1	1	Reserved	

4.1.20 UBCSA—UTILITY BUS CHIP SELECT A REGISTER

Address Offset: 4Eh
Default Value: 07h
Attribute: Read/Write
Size: 8 bits

This register enables/disables accesses to the RTC, keyboard controller, Floppy Disk controller, IDE, and BIOS locations E0000h–EFFFFh and FFF80000h–FFFDFFFFh. Disabling any of these bits prevents the encoded chip select bits (ECSADDR[2:0]) and utility bus transceiver control signal (UBUSOE#) for that device from being generated.

This register is also used to select which address range (primary or secondary) will be decoded for the resident floppy controller and IDE. This ensures that there is no contention with the Utility bus transceiver driving the system data bus during read accesses to these devices.

Bit 7: Extended BIOS Enable
When bit 7 = 1 (enabled), PCI accesses to locations FFF80000h–FFFDFFFFh result in the generation of the encoded signals (ECSADDR[2:0]) for BIOS. When enabled, PCI master accesses to this area are positively decoded and UBUSOE# is generated. When this bit is disabled (bit 7 = 0), the SIO does not generate the encoded (ECSADDR[2:0]) signals or UBUSOE#.

Bit 6: Lower BIOS Enable
When bit 6 = 1 (enabled), PCI or ISA accesses to the lower 64 KByte BIOS block (E0000h–EFFFFh) at the top of 1 MByte, or the aliases at the top of 4 GByte and 4 GByte–1 MByte results in the generation of the encoded (ECSADDR[2:0]) signals for BIOS. When enabled, PCI master accesses to this area are positively decoded to the ISA Bus, unless bit 4 in the MEMCS# Control Register is set to a 1 in which case these regions are subtractively decoded. Also, when enabled, ISA master or DMA master accesses to this region are not forwarded to the PCI Bus. When this bit is disabled (bit 6 = 0), the SIO does not generate the encoded (ECSADDR[2:0]) signals. Also, when this bit is disabled, ISA master or DMA accesses to this region are forwarded to PCI, if bit 3 in the IADCON Register is set to 1.

Bit 4: IDE Decode Enable
Bit 4 enables/disables IDE locations 1F0h–1F7h (primary) or 170h–177h (secondary) and 3F6h, 3F7h (primary) or 376h, 377h (secondary). When bit 4 = 1, the IDE encoded chip select signals and the Utility Bus transceiver signal (UBUSOE#) are generated for these addresses. When bit 4 = 0, the IDE encoded chip select signals and the Utility Bus transceiver signal (UBUSOE#) are not generated for these addresses.

Bit [5, 3:2]: Floppy Disk Address Locations Enable
Bits 2 and 3 are used to enable or disable the floppy locations as indicated below. A PCIRST# sets bit 2 to 1 and bit 3 to 0. Bit 5 is used to select between the primary and secondary address range used by the Floppy Controller and the IDE. Only primary or only secondary can be programmed at any one time. A PCIRST# sets this bit to 0 (primary).

The following table shows how these bits are used to select the floppy controller:

Address	Bit 5	Bit 3	Bit 2	DSKCHG	ECSADDR[2:0]	FLOPPYCS#
X	X	X	X	0	1 1 1	1
3F0h, 3F1h	0	1	X	1	1 0 0	0
3F2h–3F7h	0	X	1	1	1 0 0	0 (note)
370h, 371h	1	1	X	1	1 0 0	0
372h–37Fh	1	X	1	1	1 0 0	0 (note)

NOTE:
If IDE decode is enabled (bit 4 = 1), all accesses to locations 03F6h and 03F7h (primary) or 0376h and 0377h (secondary) result in the ECSADDR[2:0] signals generating a decode for IDECS1# (FLOPPYCS# is not generated). An external AND gate can be used to tie IDECS1# and FLOPPYCS# together to insure that the floppy is enabled for these accesses. If IDE decode is disabled (bit 4 = 0), and the decode for the floppy is enabled, then the encoded chip selects for the floppy locations are generated.

Bit 1: Keyboard Controller Address Location Enable
Enables (1) or disables (0) the Keyboard controller address locations 60h, 62h, 64h, and 66h. When this bit is set to 0, the Keyboard Controller encoded chip select signals (ECSADDR[2:0]) and the Utility Bus transceiver signal (UBUSOE#) are not generated for these locations.

Bit 0: RTC Address Location Enable
Enables (1) or disables (0) the RTC address locations 70h–77h. When this bit is set to 0, the RTC encoded chip select signals (ECSADDR[2:0]), RTCALE#, RTCCS#, and UBUSOE# signals are not generated for these addresses.

4.1.21 UBCSB—UTILITY BUS CHIP SELECT B REGISTER

Address Offset:	4Fh
Default Value:	4Fh
Attribute:	Read/Write
Size:	8 bits

This register is used to enable/disable accesses to the serial ports and parallel port locations supported by the SIO. When disabled, the ECSADDR(2:0) encoded chip select bits and Utility Bus Transceiver control signal (UBUSOE#), for that device, are not generated. This register is also used to disable accesses to port 92 and enable or disable configuration RAM decode.

Bit 7: Configuration RAM Decode Enable
This bit is used to enable (bit 7 = 1) or disable (bit 7 = 0) I/O write accesses to location 0C00h and I/O read/write accesses to locations 0800h–08FFh. When enabled, the encoded chip select signals for generating an external configuration page chip select (CPAGECS#) are generated for accesses to 0C00h. The encoded chip select signals for generating an external configuration memory chip select (CFIGMEMCS#) are generated for accesses to 0800h–08FFh. When bit 7 = 0, configuration RAM decode is disabled and the CPAGECS# and CFIGMEMCS# are not generated for the corresponding accesses.

Bit 6: Port 92 Enable
This bit is used to enable/disable access to Port 92. When bit 6 = 1, Port 92 is enabled. When bit 6 = 0, Port 92 is disabled. When a PCIRST# occurs, this bit is set to 1 (enable).

Bits[5:4]: Parallel Port Enable
These bits are used to select the parallel port address range: (LPT1, LPT2, LPT3, or disable). When a PCIRST# occurs, this field is set to 00 (LPT1).

Bit 5	Bit 4	Function
0	0	3BCh–3BFh (LPT1)
0	1	378h–37Fh (LP2)
1	0	278h–27Fh (LPT3)
1	1	Disabled

Bits[3:2]: Serial Port B Enable
These bits are used to assign serial port B address range: (COM1, COM2, or disable). If either COM1 or COM2 address ranges are selected, the encoded chip select signals [ECSADDR(2:0)] for Port B will be generated. A PCIRST# sets bits[3:2] to 11 (Port B disabled).

Bit 3	2	Function
0	0	3F8h–3FFh (COM1)
0	1	2F8h–2FFh (COM2)
1	0	Reserved
1	1	Port B Disabled

NOTE:
If Serial port A and B are programmed for the same I/O address, the encoded chip select signals, ECSADDR(2:0), for port B are disabled.

Bits[1:0]: Serial Port A Enable
These bits are used to assign serial port A address range: (COM1, COM2, or disable). If either COM1 or COM2 address ranges are selected, the encoded chip select signals (ECSADDR[2:0]) for Port A will be generated. A PCIRST# sets bits[1:0] to 11 (port A disabled).

Bit 1	0	Function
0	0	3F8h–3FFh (COM1)
0	1	2F8h–2FFh (COM2)
1	0	Reserved
1	1	Port A disabled

NOTE:
If Serial port A and B are programmed for the same I/O address, the encoded chip select signals, ECSADDR[2:0], for port B are disabled.

4.1.22 MAR1—MEMCS# ATTRIBUTE REGISTER #1

Address Offset: 54h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

RE—Read Enable. When the RE bit (bit 6, 4, 2, 0) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, or ISA master memory read accesses to the corresponding segment. When the RE bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory read accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master, DMA, or ISA master can not access the corresponding segment.

WE—Write Enable. When the WE bit (bit 7, 5, 3, 1) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, or ISA master memory write accesses to the corresponding segment. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory write accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master, DMA, or ISA master can not access the corresponding segment.

Bit 7: 0CC000h–0CFFFFh Exp. ROM: WE

Bit 6: 0CC000h–0CFFFFh Exp. ROM: RE

Bit 5: 0C8000h–0CBFFFh Exp. ROM: WE

Bit 4: 0C8000h–0CBFFFh Exp. ROM: RE

Bit 3: 0C4000h–0C7FFFh Exp. ROM: WE

Bit 2: 0C4000h–0C7FFFh Exp. ROM: RE

Bit 1: 0C0000h–0C3FFFh Exp. ROM: WE

Bit 0: 0C0000h–0C3FFFh Exp. ROM: RE

4.1.23 MAR2—MEMCS# ATTRIBUTE REGISTER #2

Address Offset: 55h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

RE—Read Enable. When the RE bit (bit 6, 4, 2, 0) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, or ISA master memory read accesses to the corresponding segment. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory read accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master, DMA, or ISA master can not access the corresponding segment.

WE—Write Enable. When the WE bit (bit 7, 5, 3, 1) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, or ISA master memory write accesses to the corresponding segment. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory write accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master, DMA, or ISA master can not access the corresponding segment.

Bit 7: 0DC000h–0DFFFFh Exp. ROM : WE

Bit 6: 0DC000h–0DFFFFh Exp. ROM : RE

Bit 5: 0D8000h–0DBFFFh Exp. ROM : WE

Bit 4: 0D8000h–0DBFFFh Exp. ROM : RE

Bit 3: 0D4000h–0D7FFFh Exp. ROM : WE

Bit 2: 0D4000h–0D7FFFh Exp. ROM : RE

Bit 1: 0D0000h–0D3FFFh Exp. ROM : WE

Bit 0: 0D0000h–0D3FFFh Exp. ROM : RE

4.1.24 MAR3—MEMCS# ATTRIBUTE REGISTER #3

Address Offset: 56h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

RE—Read Enable. When the RE bit (bit 6, 4, 2, 0) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, ISA master memory read accesses to the corresponding segment. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory read accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master can not access the corresponding segment.

WE—Write Enable. When the WE bit (bit 7, 5, 3, 1) is set to a 1, the SIO generates MEMCS# for PCI master, DMA, ISA master memory write accesses to the corresponding segment. When this bit is set to a 0, the SIO does not generate MEMCS# for PCI master memory write accesses to the corresponding segment. When the RE and WE bits are both 0 (or bit 4 in the MEMCS# Control Register is set to a 0–disabled), the PCI master can not access the corresponding segment.

Bit 7: 0EC000h–0EFFFFh Lower 64 KByte BIOS: WE

Bit 6: 0EC000h–0EFFFFh Lower 64 KByte BIOS: RE

Bit 5: 0E8000h–0EBFFFh Lower 64 KByte BIOS WE

Bit 4: 0E8000h–0EBFFFh Lower 64 KByte BIOS: RE

Bit 3: 0E4000h–0E7FFFh Lower 64 KByte BIOS: WE

Bit 2: 0E4000h–0E7FFFh Lower 64 KByte BIOS: RE

Bit 1: 0E0000h–0E3FFFh Lower 64 KByte BIOS: WE

Bit 0: 0E0000h–0E3FFFh Lower 64 KByte BIOS: RE

4.1.25 DMA SCATTER/GATHER RELOCATION BASE ADDRESS REGISTER

Address Offset: 57h
Default Value: 04h
Attribute: Read/Write
Size: 8 bits

The value programmed into this register determines the high order I/O address of the Scatter/Gather Command Registers, Scatter/Gather Status Registers, and the Scatter/Gather Descriptor Table Registers. The default value is 04h so the first S/G register default address is at 0410h.

Bit 7: A15

Bit 6: A14

Bit 5: A13

Bit 4: A12

Bit 3: A11

Bit 2: A10

Bit 1: A9

Bit 0: A8

4.1.26 PIRQ[3:0]#—PIRQ ROUTE CONTROL REGISTERS

Register Name: PIRQ0#, PIRQ1#, PIRQ2#, PIRQ3# Route Control

Address Offset: 60h, 61h, 62h, 63h
Default Value: 80h
Attribute: Read/Write
Size: 8 bits

These registers control the routing of PCI Interrupts (PIRQ[0:3]#) to the PC compatible Interrupts. Each PCI interrupt can be independently routed to 1 of 11 compatible interrupts. Note that two or more PCI interrupts (PIRQ[3:0]#) can be steered into the same IRQ signal (the interrupts are level sensitive and can be shared).

Each IRQ to which a PCI Interrupt is steered into must have its interrupt set to level sensitive in the Edge/Level Control Register.

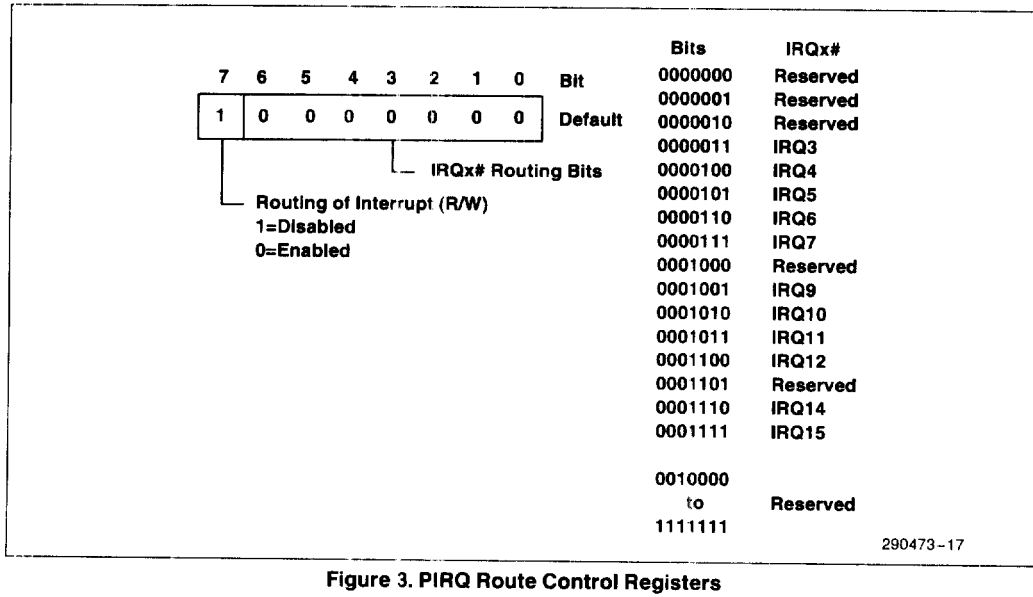


Figure 3. PIRQ Route Control Registers

Bit 7: Routing of Interrupts
When enabled, this bit routes the PCI Interrupt signal to the PC compatible interrupt signal specified in bits[3:0]. At reset, this bit is disabled (set to 1).

Bits[6:4]: Reserved
Read as 0's.

Bits[3:0]: IRQx# Routing Bits
These bits specify which IRQ signal to generate.

4.1.27 BIOS TIMER BASE ADDRESS REGISTER

Address Offset:	80h–81h
Default value:	0078h
Attribute:	Read/Write
Size:	16 bits

This register determines the base address for the BIOS Timer Register located in the I/O space. The base address can be set at Dword boundary anywhere in the 64 KByte I/O space. This register also provides the BIOS Timer access enable/disable control bit.

Bits[15:2]: BIOS Timer Base Address
Bits[15:2] correspond to PCI address lines A[15:2].

Bit 1: Reserved

Bit 0: BIOS Timer Access Enable
When bit 0 = 1, access to the BIOS Timer is enabled. When bit 0 = 0, access to the BIOS Timer is disabled. The default value is 0 (disabled).

4.1.28 SMICNTL—SMI CONTROL REGISTER

Address Offset:	A0h
Default value:	08h
Attribute:	Read/Write
Size:	8 bits

Bit 7: Reserved
Reserved for future Intel use.

Bit 6: Reserved
Reserved for future Intel use.

Bits[5:4]: Reserved
Reserved for future Intel use.

Bit 3: CTMRFRZ
Used to freeze the timers when in SMM. When this bit is set, the Fast Off timer will stop counting. This prevents time-outs from occurring while executing SMM code.

Bit 2: CSTPCLKTH
When set, the STPCLK# throttle is enabled.

Bit 1: CSTPCLKEN
When set, a read from the APMC register will cause STPCLK# to be asserted. CSTPCLKEN will be cleared by writing it to 0 or by any write to the APMC register. Enables SW to put the CPU into a low power state.

Bit 0: CSMIGATE
When this bit is written to "0" SMI# will be deasserted. When this bit is written to a "1", SMI# will be asserted if any SMIs are pending.

4.1.29 SMIEN—SMI ENABLE REGISTER

Address Offset:	A2h–A3h
Default value:	0000h
Attribute:	Read/Write
Size:	16 bits

The following bits control the enabling of associated hardware events that will generate an SMI. When set to a "1", SMI# will be asserted when the associated event occurs. When bit 7 is set, writes to the APM Control port (APMC) will generate an SMI.

Bits[15:8]: Reserved
Will be read as 0. Writes have no effect.

Bit 7: SAPMCEN
Write to APM Control Port.

Bit 6: SEXTSMIEN
EXTSMI# input asserted.

Bit 5: SFOFFTMREN
Fast Off (Idle) Timer Enable.

Bit 4: SIRQ12EN
PS/2 Mouse Interrupt.

Bit 3: SIRQ8EN
RTC Alarm Interrupt.

Bit 2: SIRQ4EN
COM2/COM4 Interrupt (Mouse).

Bit 1: SIRQ3EN
COM1/COM3 Interrupt (Mouse).

Bit 0: SIRQ1EN
Keyboard Interrupt.

4.1.30 SEE—SYSTEM EVENT ENABLE REGISTER

Address Offset: A4h, A5h, A6h, A7h
Default value: 00000000h
Attribute: Read/Write
Size: 32 bits

These bits are used to enable the corresponding hardware events as system events. When set to a "1", anytime the associated hardware event occurs, the Fast Off Timer is reloaded with its initial count. Also, when enabled the associated hardware system event is recognized as a Break Event causing STPCLK# to be deasserted.

Bit 31: FSMIEN
Prevents the system from entering Fast Off and causes STPCLK# to be deasserted when an SMI occurs.

Bit 30: Reserved
Will be read as 0. Writes have no effect

Bit 29: FNMIEN
Prevents the system from entering Fast Off and causes STPCLK# to be deasserted when an NMI occurs (parity error for example).

Bits[28:16]: Reserved
Will be read as 0. Writes have no effect.

Bits[15:3]: FIRQ[15:3]EN
This prevents the system from entering Fast Off and causes STPCLK# to be deasserted when selected hardware interrupts occur.

Bit 2: Reserved
Will be read as 0. Writes have no effect.

Bits[1:0]: FIRQ[1:0]EN
Prevents the system from entering Fast Off and causes STPCLK# to be deasserted when selected hardware interrupts occur.

4.1.31 FTMR—FAST OFF TIMER

Address Offset: A8h
Default value: 0Fh
Attribute: Read/Write
Size: 8 bits

The Fast Off Timer is used to indicate (through an SMI) that the system has been idle for a pre-programmed period of time. The Fast Off Timer consists of a count down timer that is decremented every minute. The value programmed into this register gets loaded into the Fast Off Timer when an enabled system event occurs. Each count represents one minute. When the timer expires, an SMI Special Cycle is generated. Writes to the FTMRLD register cause the Fast Off Timer to be loaded. When this register is read, the value last written to this register is returned.

PROGRAMMER'S NOTE:
Before writing to the FTMRLD register the Fast Off Timer must be stopped by writing a "1" to the CTMRFRZ bit. The Fast Off Timer will begin decrementing when the CTMRFRZ bit is subsequently set to "0".

Bits[7:0]: FTMRLD[7:0]
A write to the FTMRLD register when the Fast Off Timer is stopped (CTMRFRZ = 1) will load the Fast Off Timer with the value being written to FTMRLD register. When the Fast Off Timer is enabled (CTMRFRZ = 0) it counts down from the value loaded into it. When the Fast Off Timer reaches 00h it will trigger an SMI. If an enabled system event occurs before the Fast Off Timer reaches 00h the Fast Off Timer is reloaded with the value stored in the FTMRLD register. A read from the FTMRLD register will return the value last written to this register.

4.1.32 SMIREQ-SMI REQUEST REGISTER

Address Offset:	AAh, ABh
Default value:	00h
Attribute:	Read/Write
Size:	16 bits

These bits are status bits indicating the cause of the SMI. When an enabled event causes an SMI, the hardware automatically sets the corresponding event's request bit. The request bits are cleared by writing a "0" to them. Only the hardware can set request bits to a "1". In the event that the hardware is trying to set the bit to a "1" at the same time that it is being cleared, the hardware set to "1" will dominate.

Bits[15:8]: Reserved
Reserved for future Intel use.

Bit 7: RAPMC
When set to a "1" indicates that a write to the APM Control Port caused an SMI#.

Bit 6: REXT
When set to a "1", indicates that EXTSMI# was sampled asserted, causing an SMI#.

Bit 5: RFOFFTMR
Fast Off Timer expired causing an SMI#.

Bit 4: RIRQ12
IRQ12 was asserted causing an SMI#.

Bit 3: RIRQ8
IRQ8 was asserted causing an SMI#.

Bit 2: RIRQ4
IRQ4 was asserted causing an SMI#.

Bit 1: RIRQ3
IRQ3 was asserted causing an SMI#.

Bit 0: RIRQ1
IRQ1 was asserted causing an SMI#.

4.1.33 CTLTMRL—CLOCK THROTTLE STPCLK# LOW TIMER

Address Offset:	ACh
Default Value:	00h
Attribute:	Read/Write
Size:	8 bits

The value in this register defines the duration of the STPCLK# asserted period when the CSTPCLKTH bit is set. The value in this register is loaded into the STPCLK# Timer when STPCLK# is asserted. The STPCLK# timer runs off a 32 μs clock. Note that the timer does not begin to count until the Stop Grant Special Cycle is received.

Bits[7:0]: KSTPLOLD[7:0]
The value in this register defines the duration of the STPCLK# asserted period when the CSTPCLKTH bit is set.

4.1.34 CTLTMRH—CLOCK THROTTLE STPCLK# HIGHTIMER

Address Offset:	AEh
Default value:	00h
Attribute:	Read/Write
Size:	8 bits

The value in this register defines the duration of the STPCLK# deasserted period when the CSTPCLKTH bit is set. The value in this register is loaded into the STPCLK# Timer when STPCLK# is deasserted. The STPCLK# timer runs off a 32 μs clock.

Bits[7:0]: KSTPHILD[7:0]
The value in this register defines the duration of the STPCLK# deasserted period when the CSTPCLKTH bit is set.

4.2 DMA Register Description

The SIO contains DMA circuitry that incorporates the functionality of two 82C37 DMA controllers (DMA1 and DMA2). The DMA registers control the operation of the DMA controllers and are all accessible from the PCI Bus via PCI I/O space. In addition, some of the registers are accessed from the ISA Bus via ISA I/O space. Table 4, at the beginning of Section 4.0 lists the bus access for each register.

This section describes the DMA registers. Unless otherwise stated, a PCIRST# sets each register to its default value. The operation of the DMA is further described in Section 5.4, DMA Controller.

4.2.1 DCOM—DMA COMMAND REGISTER

Address Offset:	Channels 0-3–08h Channels 4–7–0D0h
Default Value:	00h
Attribute:	Write Only
Size:	8 bits

This 8-bit register controls the configuration of the DMA. It is programmed by the microprocessor in the Program Condition and is cleared by PCIRST# or a Master Clear instruction. Note that disabling channels 4–7 also disables channels 0–3, since channels 0–3 are cascaded onto channel 4. The DREQ and DACK# channel assertion sensitivity is assigned by channel group, not per individual channel. For priority resolution, the DMA consists of two logical channel groups—channels 0–3 (Controller 1-DMA1) and channels 4–7 (Controller 2-DMA2). Each group can be assigned fixed or rotating priority. Both groups can be assigned fixed priority, one group can be assigned fixed priority and the second rotating priority, or both groups can be assigned rotating priority. Following a PCIRST# or DMA Master Clear, both DMA1 and DMA2 are enabled in fixed priority, the DREQ sense level is active high, and the DACK# assertion level is active low.

Bit 7: DACK# Assert Level (DACK#[3:0], [7:5])
Bit 7 controls the DMA channel request acknowledge (DACK#) assertion level. Following PCIRST#, the DACK# assertion level is active low. The low level indicates recognition and acknowledgment of the DMA request to the DMA slave requesting service. Writing a 0 to bit 7 assigns active low as the assertion level. When a 1 is written to this bit, a high level on the DACK# line indicates acknowledgment of the request for DMA service to the DMA slave.

Bit 6: DREQ Sense Assert Level (DREQ[3:0], [7:5])
Bit 6 controls the DMA channel request (DREQ) assertion detect level. Following PCIRST#, the DREQ sense assert level is active high. In this condition, an active high level sampled on DREQ is decoded as an active DMA channel request. Writing a 0 to bit 6 assigns active high as the sense assert level. When a 1 is written to this bit, a low level on the DREQ line is decoded as an active DMA channel request.

Bit 5: Reserved
Must be 0.

Bit 4: DMA Group Arbitration Priority
Each channel group is individually assigned either fixed or rotating arbitration priority. At PCIRST#, each group is initialized in fixed priority. Writing a 0 to bit 4 assigns fixed priority to the channel group, while writing a 1 assigns rotating priority to the group.

Bit 3: Reserved
Must be 0.

Bit 2: DMA Channel Group Enable
Writing a 1 to this bit disables the DMA channel group, while writing a 0 to this bit enables the DMA channel group. Both channel groups are enabled following PCIRST#. Disabling channel group 4–7 also disables channel group 0–3, which is cascaded through channel 4.

Bits[1:0]: Reserved
Must be 0.

4.2.2 DCM—DMA CHANNEL MODE REGISTER

Register Name:	DMA Channel Mode
Address Offset:	Channels 0-3–0Bh Channels 4-7–0D6h
Default Value:	Bits[7:2] = 0, Bits[1:0] = undefined
Attribute:	Write Only
Size:	6 bits

Each channel has a 6-bit DMA Channel Mode Register. The Channel Mode Registers provide control over DMA Transfer type, transfer mode, address increment/decrement, and autoinitialization. Bits[1:0] select the appropriate Channel Mode Register and are not stored. Only bits[7:2] are stored in the register. This register is set to its default value upon

PCIRST# or Master Clear. Its default value is Verify transfer, Autoinitialize disable, Address increment, and Demand mode. Channel 4 defaults to cascade mode and cannot be programmed for any mode other than cascade mode.

Bits[7:6]: DMA Transfer Mode
Each DMA channel can be programmed in one of four different modes: single transfer, block transfer, demand transfer and cascade.

Bits	7	6	Transfer Mode
	0	0	Demand mode
	0	1	Single mode
	1	0	Block mode
	1	1	Cascade mode

Bit 5: Address Increment/Decrement Select
Bit 5 controls address increment/decrement during multi-byte DMA transfers. When bit 5 = 0, address increment is selected. When bit 5 = 1, address decrement is selected. Address increment is the default after a PCIRST# cycle or Master Clear command.

Bit 4: Autoinitialize Enable
When bit 4 = 1, the DMA restores the Base Page, Address, and Word count information to their respective current registers following a terminal count (TC). When bit 4 = 0, the autoinitialize feature is disabled and the DMA does not restore the above mentioned registers. A PCIRST# or Master Clear disables autoinitialization (sets bit 4 to 0).

Bits[3:2]: DMA Transfer Type
Verify, write and read transfer types are available. Verify transfer is the default transfer type upon PCIRST# or Master Clear. Write transfers move data from an I/O device to memory. Read transfers move data from memory to an I/O device. Verify transfers are pseudo transfers; addresses are generated as in a normal read or write transfer and the device responds to EOP etc. However, with Verify transfers, the ISA memory and I/O cycle lines are not driven. Bit combination 11 is illegal. When the channel is programmed for cascade ([7:6] = 11) the transfer type bits are irrelevant.

Bits	3	2	Transfer Mode
	0	0	Verify transfer
	0	1	Write transfer
	1	0	Read Transfer
	1	1	Illegal

Bits[1:0]: DMA Channel Select
Bits[1:0] select the DMA Channel Mode Register that will be written by bits[7:2].

Bits	1	0	Channel
	0	0	Channel 0 (4)
	0	1	Channel 1 (5)
	1	0	Channel 2 (6)
	1	1	Channel 3 (7)

4.2.3 DCEM—DMA CHANNEL EXTENDED MODE REGISTER

Address Offset: Channels 0-3–040Bh
Channels 4-7–04D6h
Default Value: Bits[1:0] = undefined,
Bits[3:2] = 00 for DMA1,
Bits[3:2] = 01 for DMA2,
Bits[7:4] = 0
Attribute: Write Only
Size: 6 bits

Each channel has a 6-bit Extended Mode Register. The register is used to program the DMA device data size, timing mode, EOP input/output selection, and Stop Register selection. Bits[1:0] select the appropriate Channel Extend Mode Register and are not stored. Only bits[7:2] are stored in the register. Four timing modes are available: ISA-compatible, "A", "B", and "F". Timings "A", "B", and "F" are extended timing modes and can only be run to main memory. DMA cycles to ISA expansion bus memory defaults to compatible timing if the channel is programmed in an extended timing mode.

The default bit values for each DMA group are selected upon PCIRST#. A Master Clear or any other programming sequence will not set the default register settings. The default programmed values for DMA1 channels 0-3 are 8-bit I/O count by bytes, compatible timing, and EOP output. The default values for DMA2 channels 4-7 are 16-bit I/O count by words with shifted address, compatible timing, and EOP output.

Bit 7: Reserved
Must be 0.

Bit 6: EOP Input/Output Selection
Bit 6 selects whether the EOP signal is to be used as an output during DMA transfers on this channel or an input. EOP is typically used as an output, as was available on the PC/AT. The input function was added to support data communication and other devices that would like to trigger an autoinitialize when a collision or some other event occurs. The direction of EOP is switched when DACK is changed (when a different channel is granted the bus). There may be some overlap of the SIO driving the EOP signal along with the DMA slave. However, during this overlap, both devices drive the signal to a low level (inactive). For example, assume channel 2 is about to go inactive (DACK negating) and channel 1 is about to go active. In addition, assume that channel 2 is programmed for "EOP OUT" and channel 1 is programmed for "EOP IN". When channel 2's DACK is negated and channel 1's DACK is asserted, the SIO may be driving EOP to a low value on behalf of channel 2. At the same time the device connected to channel 1 is driving EOP in to the SIO, also at an inactive level. This overlap only lasts until the SIO EOP output buffer is tri-stated, and does not effect the DMA operation. Upon PCIRST#, bit 6 is set to 0-EOP output selected.

Bits[5:4]: DMA Cycle Timing Mode
The SIO supports four DMA transfer timings: ISA-compatible, type "A", "B", and "F". Each timing and its corresponding code are described below. Upon PCIRST#, compatible timing is selected and the value of these bits is "00". The cycle timings noted below are for a SYSCLK (8.33 MHz, maximum SYSCLK frequency). DMA cycles to ISA expansion bus memory defaults to compatible timing if the channel is programmed in one of the performance timing modes (type "A", "B", or "F").

Bits[5:4] = 00: Compatible Timing
Compatible timing is provided for DMA slave devices, that, due to some design limitation, cannot support one of the faster timings. Compatible timing runs at 9 SYSCLKs (1080 nsec/single cycle) and 8 SYSCLKs (960 nsec/cycle) during the repeated portion of a BLOCK or DEMAND mode transfer.

Bits[5:4] = 01: Type "A" Timing
Type "A" timing is provided to allow shorter cycles to main memory (via the PCI Bus). Type "A" timing runs at 6 SYSCLKs (720 nsec/cycle) during the repeated portion of a BLOCK or DEMAND mode transfer. Type "A" timing varies from compatible timing primarily in shortening the memory operation to the minimum allowed main memory. The I/O portion of the cycle (data setup on write, I/O read access time) is the same as with compatible cycles. The actual active command time is shorter. However, it is expected that the DMA devices that provide the data access time or write data setup time should not require excess IOR# or IOW# command active time. Because of this, most ISA DMA devices should be able to use type "A" timing.

Bits[5:4] = 10: Type "B" Timing
Type "B" timing is provided for 8/16-bit ISA DMA devices that can accept faster I/O timing. Type "B" only works with fast main memory. Type "B" timing runs at 5 SYSCLKs (600 nsec/cycle) during the repeated portion of a BLOCK or DEMAND mode transfer. Type "B" timing requires faster DMA slave devices than compatible timing. In Type "B" timing the cycles are shortened so that the data setup time on I/O write cycles is shortened and the I/O read access time is required to be faster.

Bits[5:4] = 11: Type "F" Timing
Type "F" timing provides high performance DMA transfer capability. Type "F" timing runs at 3 SYSCLKs (360 nsec/single cycle) during the repeated portion of a BLOCK or DEMAND mode transfer, resulting in a maximum data transfer rate of 8.33 MBytes/second.

Bits[3:2]: Addressing Mode
The SIO supports both 8- and 16-bit DMA device data sizes. Three data size options are programmable with bits[3:2]. Both the 8-bit I/O, "count by bytes" mode and the 16-bit I/O, "count by words" (address shifted) mode are ISA compatible. The 16-bit I/O, "count by bytes" mode is offered as an extension of the ISA compatible modes. Bits[3:2] = 10 is reserved. Byte assembly/disassembly is performed by the ISA control unit. Each of the data transfer size modes is discussed below.

Bits[3:2] = 00: 8-bit I/O, "Count By Bytes" Mode
In 8-bit I/O, "count by bytes" mode, the Current Address Register can be programmed to any address. The Current Byte/Word Count Register is programmed with the "number of bytes minus 1" to transfer.

Bits[3:2] = 01: 16-bit I/O, "Count By Words" (Address Shifted) Mode
In "count by words" mode (address shifted), the Current Address Register can be programmed to any even address, but must be programmed with the address value shifted right by one bit. The Low Page and High Page Registers are not shifted during DMA transfers. Thus, the least significant bit of the Low Page register is ignored when the address is driven out onto the bus. The Current Byte/Word Count Register is programmed with the number of words minus 1 to be transferred.

Bits[3:2] = 10: Reserved

Bits[3:2] = 11: 16-Bit I/O, "Count By Bytes" Mode
In 16-bit "count by bytes" mode, the Current Address Register can be programmed to any byte address. For most DMA devices, however, it should be programmed only to even addresses. If the address is programmed to an odd address, the DMA controller does a partial word transfer during the first and last transfer, if necessary. The bus controller does the Byte/Word assembly necessary to write any size memory device. In this mode, the Current Address Register is incremented or decremented by two and the byte count is decremented by the number of bytes transferred during each bus cycle. The Current Byte/Word Count Register is programmed with the "number of bytes minus 1" to be transferred. This mode should only be programmed for 16-bit ISA DMA slaves.

Bits[1:0]: DMA Channel Select
Bits[1:0] select the particular channel that will have its DMA Channel Extend Mode Register programmed with bits[7:2].

Bits	1	0	Channel
	0	0	Channel 0 (4)
	0	1	Channel 1 (5)
	1	0	Channel 2 (6)
	1	1	Channel 3 (7)

4.2.4 DR—DMA REQUEST REGISTER

Address Offset:	Channels 0-3--09h Channels 4-7--0D2h
Default Value:	Bits[1:0] = undefined, Bits[7:2] = 0
Attribute:	Write Only
Size:	4 bits

Each channel has a request bit in one of the two 4-bit DMA Request Registers. The Request Register is used by software to initiate a DMA request. The DMA responds to the software request as though DREQ[x] is asserted. These requests are non-maskable and subject to prioritization by the priority encoder network. Each register bit is set to 1 or 0 separately under software control or is set to 0 upon generation of a TC. The entire register is set to 0 upon PCIRST# or a Master Clear. It is not affected upon a RSTDRV output. To program a bit, the software loads the proper form of the data word. Bits[1:0] determine which channel Request Register will be written. In order to make a software request, the channel must be in Block Mode. The Request Register status for DMA1 and DMA2 is output on bits[7:4] of a Status Register read to the appropriate port.

Bits[7:3]: Reserved
Must be 0.

Bit 2: DMA Channel Service Request
Writing a 0 to bit 2 resets the individual software DMA channel request bit. Writing a 1 to bit 2 sets the request bit. The request bit for each DMA channel is reset to 0 upon a PCIRST# or a Master Clear.

Bits[1:0]: DMA Channel Select
Bits[1:0] select the DMA channel mode register to program with bit 2.

Bits	1	0	Channel
	0	0	Channel 0
	0	1	Channel 1 (5)
	1	0	Channel 2 (6)
	1	1	Channel 3 (7)

4.2.5 MASK REGISTER—WRITE SINGLE MASK BIT

Address Offset: Channels 0-3–0Ah
Channels 4-7–0D4h
Default Value: Bits[1:0] = undefined, Bit 2 = 1, Bits[7:3] = 0
Attribute: Write Only
Size: 1 bit/channel

Each DMA channel has a mask bit that enables/disables an incoming DMA channel service request DREQ[x]. Two 4-bit registers store the current mask status for DMA1 and DMA2. Setting the mask bit disables the incoming DREQ[x] for that channel. Clearing the mask bit enables the incoming DREQ[x]. A channel's mask bit is automatically set when the Current Byte/Word Count register reaches terminal count (unless the channel is programmed for autoinitialization). Each mask bit may also be set or cleared under software control. The entire register is also set by a PCIRST# or a Master Clear. Setting the entire register disables all DMA requests until a clear mask register instruction allows them to occur. This instruction format is similar to the format used with the DMA Request Register.

Individually masking DMA channel 4 (DMA controller 2, channel 0) will automatically mask DMA channels [3:0], as this channel group is logically cascaded onto channel 4. Setting this mask bit disables the incoming DREQ's for channels [3:0].

Bits[7:3]: Reserved
Must be 0.

Bit 2: Channel Mask Select
When bit 2 is set to a 1, DREQ is disabled for the selected channel. When bit 2 is set to a 0, DREQ is enabled for the selected channel.

Bit[1:0]: DMA Channel Select
Bits[1:0] select the DMA Channel Mode Register to program with bit 2.

Bits	1	0	Channel
	0	0	Channel 0 (4)
	0	1	Channel 1 (5)
	1	0	Channel 2 (6)
	1	1	Channel 3 (7)

4.2.6 MASK REGISTER—WRITE ALL MASK BITS

Address Offset: Channels 0-3–0Fh
Channels 4-7–0DEh
Default Value: Bit[3:0] = 1, Bit[7:4] = 0
Attribute: Read/Write
Size: 4 bits

Writing to this register enables/disables incoming DREQ assertions. There are four mask bits per register, one for each channel. This permits all four channels to be simultaneously enabled/disabled instead of enabling/disabling each channel individually, as is the case with the Mask Register—Write Single Mask Bit.

Two 4-bit registers store the current mask status for DMA1 and DMA2. Unlike the Mask Register—Write Single Mask Bit, this register and includes a status read to check the current mask status of the selected DMA channel group. A channel's mask bit is automatically set to 1 when the Current Byte/Word Count Register reaches terminal count (unless the channel is programmed for autoinitialization). Bits[3:0] are set to 1 by a PCIRST# or a Master Clear. Setting bits[3:0] to 1 disables all DMA requests until a clear mask register instruction enables the requests.

Two important points should be taken into consideration when programming the mask registers. First, individually masking DMA channel 4 (DMA controller 2, channel 0) will automatically mask DMA channels [3:0], as this channel group is logically cascaded onto channel 4. Second, masking DMA controller 2 with a write to port 0DEh will also mask DREQ assertions from DMA controller 1 for the same reason. When DMA channel 4 is masked, so are DMA channels 0-3.

Bits[7:4]: Reserved
Must be 0.

Bits[3:0]: Channel Mask Bits
Setting the bit(s) to a 1 disables the corresponding DREQ(s). Setting the bit(s) to a 0 enables the corresponding DREQ(s). Bits[3:0] are set to 1 upon PCIRST# or Master Clear. When read, bits[3:0] indicate the DMA channel [3:0] ([7:4]) mask status.

Bit	Channel
0	0 (4)
1	1 (5)
2	2 (6)
3	3 (7)

NOTE:
Disabling channel 4 also disables channels 0-3 due to the cascade of DMA1 through channel 4 of DMA2.

4.2.7 DS—DMA STATUS REGISTER

Address Offset:	Channels 0-3–08h
	Channels 4-7–0D0h
Default Value:	00h
Attribute:	Read Only
Size:	8 bits

Each DMA controller has a read-only DMA Status Register. This register indicates which channels have reached terminal count and which channels have a pending DMA request. Bits[3:0] are set every time the corresponding TC is reached by that channel. Bits[3:0] are set to 0 upon PCIRST# and on each status read. Bits[7:4] are set whenever their corresponding channel is requesting service.

Bits[7:4]: Channel Request Status
When a valid DMA request is pending for a channel (on its DREQ signal line), the corresponding bit is set to 1. When a DMA request is not pending for a particular channel, the corresponding bit is set to 0. The source of the DREQ may be hardware, a timed-out block transfer, or a software request. Note that channel 4 does not have DREQ or DACK lines, so the response for a read of DMA2 status for channel 4 is irrelevant.

Bit	Channel
4	0
5	1 (5)
6	2 (6)
7	3 (7)

Bits[3:0]: Channel Terminal Count Status
When a channel reaches terminal count (TC), its status bit is set to 1. If TC has not been reached, the status bit is set to 0. Note that channel 4 is programmed for cascade, and is not used for a DMA transfer. Therefore, the TC bit response for a status read on DMA2 for channel 4 is irrelevant.

Bit	Channel
0	0
1	1 (5)
2	2 (6)
3	3 (7)

4.2.8 DMA BASE AND CURRENT ADDRESS REGISTERS (8237 COMPATIBLE SEGMENT)

Address Offset:	DMA Channel 0–000h
	DMA Channel 1–002h
	DMA Channel 2–004h
	DMA Channel 3–006h
	DMA Channel 4–0C0h
	DMA Channel 5–0C4h
	DMA Channel 6–0C8h
	DMA Channel 7–0CCh
Default Value:	All bits undefined
Attribute:	Read/Write
Size:	16 bits per channel

Each channel has a 16-bit Current Address Register. This register contains the value of the 16 least significant bits of the full 32-bit address used during DMA transfers. The address is automatically incremented or decremented after each transfer and the intermediate values of the address are stored in the Current Address Register during the transfer. This register is written to or read from by the PCI Bus or ISA Bus master in successive 8-bit bytes. The programmer must issue the "Clear Byte Pointer Flip-Flop" command to reset the internal byte pointer and correctly align the write prior to programming the Current Address Register. After clearing the Byte Pointer Flip-Flop, the first write to the Current Address Register programs the low byte, bits[7:0], and the second write programs the high byte, bits[15:8]. This procedure also applies to read cycles. It may also be reinitialized by an Autoinitialize back to its original value. Autoinitialize takes place only after a TC or EOP.

Each channel has a Base Address Register located at the same port address as the corresponding Current Address Register. These registers store the original value of their associated Current Address Registers. During autoinitialize these values are used to restore the Current Address Registers to their original values. The Base Registers are written simultaneously with their corresponding Current Address Register in successive 8-bit bytes. The Base Registers are write-only.

In Scatter/gather mode, these registers store the lowest 16 bits of the current memory address. During a Scatter/gather transfer, the DMA will load a reserve buffer into the base memory address register.

Bits[15:0]: Base and Current Address [15:0]
These bits represent the 16 least significant address bits used during DMA transfers. Together with the DMA Low Page Register, they form the ISA-compatible 24-bit DMA address. As an extension of the ISA compatible functionality, the DMA High Page Register completes the 32-bit address needed when implementing SIO extensions such as DMA to the PCI Bus slaves that can take advantage of full 32-bit addressability. Upon PCIRST# or Master Clear, the value of these bits is 0000h.

4.2.9 DMA BASE AND CURRENT BYTE/WORD COUNT REGISTERS (8237 COMPATIBLE SEGMENT)

Address Offset:	DMA Channel 0–001h
	DMA Channel 1–003h
	DMA Channel 2–005h
	DMA Channel 3–007h
	DMA Channel 4–0C2h
	DMA Channel 5–0C6h
	DMA Channel 6–0CAh
	DMA Channel 7–0CEh
Default Value:	All bits undefined
Attribute:	Read/Write
Size:	16 bits per channel

Each channel has a 16-bit Current Byte/Word Count Register. This register determines the number of transfers to be performed. The actual number of transfers is one more than the number programmed in the Current Byte/Word Count Register (i.e., programming a count of 100 results in 101 transfers). The Byte/Word count is decremented after each transfer. The intermediate value of the Byte/Word count is stored in the register during the transfer. When the value in the register goes from zero to 0FFFFh, a TC is generated.

Following the end of a DMA service the register may also be re-initialized by an autoinitialization back to its original value. Autoinitialize can only occur when a TC occurs. If it is not autoinitialized, this register has a count of FFFFh after TC.

When the Extended Mode Register is programmed for, or defaulted to, transfers to/from an 8-bit I/O, the Byte/Word count indicates the number of bytes to be transferred.

When the Extended Mode Register is programmed for, or defaulted to, transfers to/from a 16-bit I/O, with shifted address, the Byte/Word count indicates the number of 16-bit words to be transferred.

When the Extended Mode Register is programmed for transfers to/from a 16-bit I/O, the Byte/Word Count indicates the number of bytes to be transferred. The number of bytes does not need to be a multiple of two or four in this case.

Each channel has a Base Byte/Word Count Register located at the same port address as the corresponding Current Byte/Word Count Register. These registers store the original value of their associated Current Byte/Word Count Registers. During Autoinitialize these values are used to restore the Current registers to their original values. The Base registers are written simultaneously with their corresponding Current register in successive 8-bit bytes. The Base registers cannot be read by any external agents.

In Scatter/gather mode, these registers store the 16 bits of the current Byte/Word count. During Scatter/gather transfer, the DMA will load a reserve buffer into the base Byte/Word Count register.

Bits[15:0]: Base and Current Byte/Word Count
These bits represent the 16 byte/word count bits used when counting down a DMA transfer. Upon PCIRST# or Master Clear, the value of these bits is 0000h.

4.2.10 DMA MEMORY BASE LOW PAGE AND CURRENT LOW PAGE REGISTERS

Register Name:	DMA Memory Current Low Page Register (Read/Write) DMA Memory Base Low Page Register (Write Only)
Address Offset:	DMA Channel 0–087h DMA Channel 1–083h DMA Channel 2–081h DMA Channel 3–082h DMA Channel 5–08Bh DMA Channel 6–089h DMA Channel 7–08Ah
Default Value:	All bits undefined
Size:	8 bits per channel

Each channel has an 8-bit Low Page Register. The DMA memory Low Page Register contains the eight second most-significant bits of the 32-bit address. The register works in conjunction with the DMA controller's High Page Register and Current Address Register to define the complete (32-bit) address for the DMA channel. This 8-bit register is read or written directly. It may also be re-initialized by an autoinitialize back to its original value. Autoinitialize takes place only after a TC or EOP.

Each channel has a Base Low Page Address Register located at the same port address as the corresponding Current Low Page Register. These registers store the original value of their associated Current Low Page Registers. During autoinitialization, these values are used to restore the Current Low Page Registers to their original values. The 8-bit Base Low Page Registers are written simultaneously with their corresponding Current Low Page Register by the microprocessor. The Base Low Page registers are write only.

During Scatter/gather, these registers store the 8 bits from the third byte of the current memory address. During a Scater-Gather transfer, the DMA will load a reserve buffer into the base memory address register.

Bits[7:0]: DMA Low Page and Base Low Page [23:16]
These bits represent the eight second most significant address bits when forming the full 32-bit address for a DMA transfer. Upon PCIRST# or Master Clear, the value of these bits is 00h.

4.2.11 DMA MEMORY BASE HIGH PAGE AND CURRENT HIGH PAGE REGISTERS

Register Name:	DMA Memory Current High Page Register (Read/Write) DMA Memory Base High Page Register (Write Only)
Address Offset:	DMA Channel 0–0487h DMA Channel 1–0483h DMA Channel 2–0481h DMA Channel 3–0482h DMA Channel 5–048Bh DMA Channel 6–0489h DMA Channel 7–048Ah
Default Value:	All bits undefined
Size:	8 bits per channel

Each channel has an 8-bit Current High Page Register. The DMA memory Current High Page Register contains the eight most significant bits of the 32-bit address. The register works in conjunction with the DMA controller's Current Low Page Register and Current Address Register to define the complete (32-bit) address for the DMA channels and corresponds to the Current Address Register for each channel. This 8-bit register is read or written directly. It may also be autoinitialized back to its original value. Autoinitialize takes place only after a TC or EOP.

This register is set to 0 during the programming of both the Current Low Page Register and the Current Address Register. Thus, if this register is not programmed after the other address and Low Page Registers are programmed, then its value is 00h. In this case, the DMA channel operates the same as an 82C37 (from an addressing standpoint). This is the address compatibility mode.

If the high 8 bits of the address are programmed after the other addresses, then the channel modifies its operation to increment (or decrement) the entire 32-bit address. This is unlike the 82C37 "Page" register in the original PCs which could only increment to a 64 KByte boundary for 8-bit channels or 128 KByte boundary for 16-bit channels. This is extended address mode. In this mode, the ISA Bus controller generates the signals MEMR# and MEMW# only for addresses below 16 MBytes.

Each channel has a Base High Page Register located at the same port address as the corresponding Current High Page Register. These registers store the original value of their associated Current High Page Registers. During autoinitialize, these values are used to restore the Current High Page Registers to their original values. The 8-bit Base High Page Registers are written simultaneously with their corresponding Current High Page Register. The Base High Page Registers are write only.

During Scatter/Gather, these registers store the 8 bits from the highest byte of the current memory address. During a Scatter/Gather transfer, the DMA will load a reserve buffer into the base memory address register.

Bits[7:0]: DMA High Page and Base High Page [31:24]
These bits represent the eight most-significant address bits when forming the full 32-bit address for a DMA transfer. Upon PCIRST# or Master Clear, the value of these bits is 00h.

4.2.12 DMA CLEAR BYTE POINTER REGISTER

Address Offset:	Channels 0–3–00Ch
	Channels 4–7–0D8h
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bits

Writing to this register executes the clear byte pointer command. This command is executed prior to writing or reading new address or word count information to the DMA. This command initializes the byte pointer flip-flop to a known state so that subsequent accesses to register contents will address upper and lower bytes in the correct sequence.

The clear byte pointer command clears the internal latch used to address the upper or lower byte of the 16-bit Address and Word Count Registers. The latch is also cleared at power on by PCIRST# and by the Master Clear command. The Host CPU may read or write a 16-bit DMA controller register by performing two consecutive accesses to the I/O port. The Clear Byte Pointer command precedes the first access. The first I/O write to a register port loads the least significant byte, and the second access automatically accesses the most significant byte.

When DMA registers are being read or written, two Byte Pointer flip-flops are used. One flip-flop is for Channels 0–3 and one for Channels 4–7. Both of these act independently. There are separate software commands for clearing each of them (0Ch for Channels 0–3, 0D8h for Channels 4–7).

Bits[7:0]: Clear Byte Pointer
No specific pattern. Command enabled with a write to the I/O port address.

4.2.13 DMC—DMA MASTER CLEAR REGISTER

Address Offset:	Channel 0–3–00Dh
	Channel 4–7–0DAh
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bit

This software instruction has the same effect as the hardware Reset. The Command, Status, Request, and Internal First/Last Flip-Flop registers are cleared and the Mask Register is set. The DMA controller enters the idle cycle. There are two independent Master Clear Commands; 0Dh acts on Channels 0–3, and 0DAh acts on Channels 4–7.

Bits[7:0]: Master Clear
No specific pattern. Command enabled with a write to the I/O port address.

4.2.14 DCM—DMA CLEAR MASK REGISTER

Address Offset:	Channel 0–3–00Eh
	Channel 4–7–0DCh
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bit

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

Bits[7:0]: Clear Mask Register
No specific pattern. Command enabled with a write to the I/O port address.

4.2.15 SCATTER/GATHER COMMAND REGISTER

Register Name: DMA Scatter Gather Command

Address Offset: Channels 0 default address—0410h
Channels 1 default address—0411h
Channels 2 default address—0412h
Channels 3 default address—0413h
Channels 5 default address—0415h
Channels 6 default address—0416h
Channels 7 default address—0417h

Default Value: 00h
Attribute: Write Only, Relocatable
Size: 8 bits

The Scatter/Gather Command Register controls operation of the descriptor table aspect of scatter/gather transfers. This register can be used to start and stop a scatter/gather transfer. The register can also be used to select between IRQ13 and EOP to be asserted following a terminal count. The current scatter/gather transfer status can be read in the scatter/gather channel's corresponding Scatter/Gather Status Register. After a PCIRST# or Master Clear, IRQ13 is disabled and EOP is enabled.

Bit 7: IRQ13/EOP Select
Bit 7, if enabled via bit 6 of this register, selects whether EOP or IRQ13 is asserted at termination caused by a last buffer expiring. The last buffer can be either the last buffer in the list or the last buffer loaded in the DMA while it is suspended. If bit 7 = 1 (and bit 6 = 1), EOP is asserted when the last buffer is completed. If bit 7 = 0 (and bit 6 = 1), IRQ13 is asserted when the last buffer is completed.

EOP can be used to alert an expansion bus I/O device that a scatter/gather termination condition was reached. The I/O device, in turn, can assert its own interrupt request line to invoke a dedicated interrupt handling routine. IRQ13 should be used when the CPU needs to be notified directly.

Following PCIRST#, or Master Clear, the value stored for this bit is "1", and EOP is selected. Bit-6 must be set to a "1" to enable this bit during a S/G Command register write. When bit 6 is a "0" during the write, bit 7 will not have any effect on the current EOP/IRQ13 selection.

Bit 6: IRQ13/EOP Programming Enable
Enabling IRQ13/EOP programming allows initialization or modification of the S/G termination handling bits. When bit 6 = 0, bit 7 does not affect the state of IRQ13 or EOP assertion. When bit 6 = 1, bit 7 determines the termination handling following a terminal count.

Bits[5:2]: Reserved
Must be 0.

Bits[1:0]: Scatter/Gather Commands
This 2-bit field is used to start and stop scatter/gather.

Bits[1:0] = 00: No S/G operation
No S/G command operation is performed. Bits[7:6] may still be used to program IRQ13/EOP selection.

Bits[1:0] = 01: Start S/G Command
The Start command initiates the scatter/gather process. Immediately after the start command is issued (setting bits[1:0] to 01), a request is issued to fetch the initial buffer from the descriptor table to fill the Base Register set in preparation for performing a transfer. The buffer prefetch request has the same priority with respect to other channels as the DREQ it is associated with. Within the channel, DREQ is higher in priority than a prefetch request.

The Start command assumes the Base and Current registers are both empty and will request a prefetch automatically. Note that this command also sets the Scatter/Gather Status Register to S/G Active, Base Empty, Current Empty, not Terminated, and Next Null Indicator to 0. The EOP/IRQ13 bit will still reflect the last value programmed.

Bits[1:0] = 10: Stop S/G Command
The Stop command halts a Scatter/gather transfer immediately. When a Stop command is given, the Terminate bit in the S/G Status register and the DMA channel mask bit are both set.

Bits[1:0] = 11: Reserved

4.2.16 SCATTER/GATHER STATUS REGISTER

Address Offset:	Channels 0 default address—0418h Channels 1 default address—0419h Channels 2 default address—041Ah Channels 3 default address—041Bh Channels 5 default address—041Dh Channels 6 default address—041Eh Channels 7 default address—041Fh
Default Value:	00h
Attribute:	Read Only, Relocatable
Size:	8-bits

The Scatter/Gather Status Register contains information on the scatter/gather transfer status. This register provides dynamic status information on S/G transfer activity, the current and base buffer state, S/G transfer termination, and the End of the List indicator.

An Active bit is set to "1" after the S/G Start command is issued. The Active bit will be "0" before the initial Start command, following a terminal count, and after a S/G Stop command is issued. The Current Register and Base Register Status bits indicate whether the corresponding register has a buffer loaded. It is possible for the Base Register Status to be set while the Current Register Status is cleared. When the Current Register transfer is complete, the Base Register will not be moved into the Current Register until the start of the next data transfer. Thus, the Current Register State is empty (cleared), while the Base Register State is full (set). The Terminate bit is set active after a Stop command, after TC for the last buffer in the list, and both Base and Current Registers have expired. The EOP and IRQ13 bits indicate which end of process indicator will be used to alert the system of an S/G process termination. The EOL status bit is set if the DMA controller has loaded the last buffer of the Link List. Following PCIRST#, or Master Clear, each bit in this register is reset to "0".

Bit 7: Next Link Null Indicator
If the next scatter/gather descriptor fetched from memory during a fetch operation has the EOL value set to 1, the current value of the Next Link Register is not overwritten. Instead, bit 7 of the channel's Scatter/Gather Status Register is set to a 1. If the fetch returns a EOL value set to 0, this bit is set to 0. This status bit is written after every fetch operation. Following PCIRST#, or Master Clear, this bit is set to 0. This bit is also cleared by an S/G Start Command write to the Scatter/Gather Command Register.

Bit 6: Reserved

Bit 5: Issue IRQ13/EOP on Last Buffer
When bit 5 = 0, EOP was either defaulted to at reset or selected through the Scatter/Gather Command Register as the S/G process termination indicator. EOP is issued when a terminal count occurs or following the Stop Command. When bit 5 = 1, an IRQ13 is issued to alert the CPU of this same status.

Bit 4: Reserved

Bit 3: Scatter/Gather Base Register Status
When bit 3 = 0, the Base Register is empty. When bit 3 = 1, the Base Register has a buffer link loaded. Note that the Base Register State may be set while the Current Register state is cleared. This condition occurs when the Current Register expires following a transfer. The Base Register will not be moved into the Current Register until the start of the next DMA transfer.

Bit 2: Scatter/Gather Current Register Status
When bit 2 = 0, the Current Register is empty. When bit 2 = 1, the Current Register has a buffer link loaded and is considered full. Following PCIRST#, bit 2 is set to 0.

Bit 1: Reserved

Bit 0: Scatter/Gather Active
The Scatter/gather Active bit indicates the current S/G transfer status. Bit 0 is set to a 1 after an S/G Start Command is issued. Bit 0 is set to 0 before the Start Command is issued. Bit 0 is 0 after terminal count on the last buffer on the channel is reached. Bit 0 is also 0 after an S/G Stop Command has been issued. Following a PCIRST# or Master Clear, this bit is 0.

4.2.17 SCATTER/GATHER DESCRIPTOR TABLE POINTER REGISTER

Address Offset: Channel 0 default address—0420h–0423h
Channel 1 default address—0424h–0424h
Channel 2 default address—0428h–042Bh
Channel 3 default address—042Ch–042Ch
Channel 5 default address—0434h–0437h
Channel 6 default address—0438h–043Bh
Channel 7 default address—043Ch–043Fh
Default Value: All bits undefined
Attribute: Read/Write, Relocatable
Size: 32 bits

The Scatter/Gather Descriptor Table Pointer Register contains the 32-bit pointer address to the first scatter/gather descriptor entry in the descriptor table in memory. Before the start of a S/G transfer, this register should be programmed to point to the first descriptor in the Scatter/Gather Descriptor Table. Following a S/G Start command, the SIO reads the first SGD entry. Subsequently, at the end of the each buffer block transfer, the contents of the SGD Table pointer registers are incremented by 8 until the end of the SGD Table is reached.

The Scatter/Gather Descriptor Table Pointer Registers can be programmed with a single 32-bit PCI write.

Following a prefetch to the address pointed to by the channel's Scatter/Gather Descriptor Table Pointer Register, the new memory address is loaded into the Base Address Register, the new Byte Count is loaded into the Base Byte Count Register, and the newly fetched next scatter/gather descriptor replaces the current next scatter/gather value.

The end of the Scatter/Gather Descriptor Table is indicated by an End of Table field having a MSB equal to 1. When this value is read during a scatter/gather descriptor fetch, the current scatter/gather descriptor value is not replaced. Instead, bit 7 of the channel's Status Register is set to a 1, when the EOL is read from memory.

Bits[31:0]:
The Scatter/Gather Descriptor Table Pointer Register contains a 32-bit pointer address to the main memory location where the software maintains the Scatter Gather Descriptors for the linked-list buffers. Bits[31:0] correspond to A[31:0] on the PCI.

4.2.18 SCATTER/GATHER INTERRUPT STATUS REGISTER

Address Offset: 040Ah
Default Value: 00h
Attribute: Read Only, Relocatable
Size: 8 bits

The Scatter/Gather Interrupt Status Register is a read only register and is used to indicate the source (channel) of a DMA Scatter/Gather interrupt on IRQ13. The DMA controller drives IRQ13 active after reaching terminal count during a Scatter/Gather transfer. It does not drive IRQ13 active during the initial programming sequence that loads the Base registers.

Bit 7: Channel 7 Interrupt Status
When this bit is set to a 1, Channel 7 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 6: Channel 6 Interrupt Status
When this bit is set to a 1, Channel 6 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 5: Channel 5 Interrupt Status
When this bit is set to a 1, Channel 5 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 4: Reserved
Read as 0.

Bit 3: Channel 3 Interrupt Status
When this bit is set to a 1, Channel 3 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 2: Channel 2 Interrupt Status
When this bit is set to a 1, Channel 2 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 1: Channel 1 Interrupt Status
When this bit is set to a 1, Channel 1 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

Bit 0: Channel 0 Interrupt Status
When this bit is set to a 1, Channel 0 has an interrupt due to a Scatter/Gather Transfer; otherwise this bit is set to a 0.

4.3 Timer Register Description

The SIO contains three counters that are equivalent to those found in the 82C54 Programmable Interval Timer. The Timer registers control these counters and can be accessed from either the ISA Bus via ISA I/O space or the PCI Bus via PCI I/O space.

This section describes the counter/timer registers on the SIO. The counter/timer operations are further described in Section 5.7, Timer Unit.

4.3.1 TCW—TIMER CONTROL WORD REGISTER

Address Offset: 043h
Default Value: All bits undefined
Attribute: Write Only
Size: 8 bits

The Timer Control Word Register specifies the counter selection, the operating mode, the counter byte programming order and size of the count value, and whether the counter counts down in a 16-bit or binary-coded decimal (BCD) format. After writing the control word, a new count can be written at any time. The new value will take effect according to the programmed mode.

There are six programmable counting modes. Typically, the SIO Timer Counters 0 and 2 are programmed for Mode 3, the Square Wave Mode, while Counter 1 is programmed in Mode 2, the Rate Generator Mode.

Two special commands are selected through the Timer Control Word Register. The Read Back Command (see Section 4.3.1.1) is selected when bits[7:6] are both 1 and the Counter Latch Command (see Section 4.3.1.2) is selected when bits[5:4] are both 0. When either of these two commands are selected, the meaning of the other bits in the register changes.

Bits 4 and 5 are also used to select the count register programming mode. The read/write selection chosen with the control word indicates the programming sequence that must follow when initializing the specified counter. If a counter is programmed to read/write two byte counts, note that a program must not transfer control between writing the first and second byte to another routine that also writes into that same counter. Otherwise, the counter will be loaded with an incorrect count. The count must always be completely loaded with both bytes.

Bits 6 and 7 are also used to select the counter for the control word being written.

Following PCIRST#, the control words for each register are undefined. Each timer must be programmed to bring it into a known state. However, each counter OUT signal is set to 0 following PCIRST#. The SPKR output, interrupt controller input IRQ0 (internal), bit 5 of port 061h, and the internally generated refresh request are each set to 0 following PCIRST#.

Bits[7:6]: Counter Select
The Counter Selection bits select the counter the control word acts upon as shown below. The Read Back Command is selected when bits[7:6] are both 1.

Bit	7	6	Function
	0	0	Counter 0 select
	0	1	Counter 1 select
	1	0	Counter 2 select
	1	1	Read Back Command (see Section 4.3.1.1)

Bits[5:4]: Read/Write Select
Bits[5:4] are the read/write control bits. The Counter Latch Command is selected when bits[5:4] are both 0. The read/write options include r/w least significant byte, r/w most significant byte, or r/w the LSB and then the MSB. The actual counter programming is done through the counter I/O port (040h, 041h, and 042h for counters 0, 1, and 2, respectively).

Bit	5	4	Function
	0	0	Counter Latch Command (see Section 4.3.1.2)
	0	1	R/W Least Significant Byte (LSB)
	1	0	R/W Most Significant Byte (MSB)
	1	1	R/W LSB then MSB

Bits[3:1]: Counter Mode Selection
Bits[3:1] select one of six possible modes of operation for the counter as shown below.

Bit 3	2	1	Mode	Function
0	0	0	0	Out signal on end of count (= 0)
0	0	1	1	Hardware retriggerable one-shot
X	1	0	2	Rate generator (divide by n counter)
X	1	1	3	Square wave output
1	0	0	4	Software triggered strobe
1	0	1	5	Hardware triggered strobe

Bit 0: Binary/BCD Countdown Select
When bit 0 = 0, a binary countdown is used. The largest possible binary count is 2^{16}. When bit 0 = 1, a binary coded decimal (BCD) count is used. The largest BCD count allowed is 10^4.

4.3.1.1 Read Back Command

The Read Back Command is used to determine the count value, programmed mode, and current states of the OUT pin and Null count flag of the selected counter or counters. The Read Back Command is written to the Timer Control Word Register which latches the current states of the above mentioned variables. The value of the counter and its status may then be read by I/O access to the counter address.

Status and/or count may be latched on one, two, or all three of the counters by selecting the counter during the register write. The count latched remains latched until read, regardless of further latch commands. The count must be read before newer latch commands latch a new count. The status latched by the Read Back Command also remains latched until after a read to the counter's I/O port by reading the Counter Access Ports Register. Thus, the status and count are unlatched only after a counter read of the Timer Status Byte Format Register, the Counter Access Ports Register, or the Timer Status Byte Register and Counter Access Ports Register in succession.

Both count and status of the selected counter(s) may be latched simultaneously by setting both bit 5 and bit 4 to 0. This is functionally the same as issuing two consecutive, separate Read Back Commands. As mentioned above, if multiple count and/or status Read Back Commands are issued to the same counter(s) without any intervening reads, all but the first are ignored.

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

NOTE:
The Timer Counter Register bit definitions are different during the Read Back Command than for a normal Timer Counter Register write.

Bits[7:6]: Read Back Command
When bits[7:6] are both 1, the Read Back Command is selected during a write to the Timer Control Word Register. As noted above, the normal meanings (mode, countdown, r/w select) of the bits in the control register at I/O address 043h change when the Read Back Command is selected. Following the Read Back Command, I/O reads from the selected counter's I/O addresses produce the current latch status, the current latched count, or both if bits 4 and 5 are both 0.

Bit 5: Latch Count of Selected Counters
When bit 5 = 1, the current count value of the selected counters will be latched. When bit 4 = 0, the status will not be latched.

Bit 4: Latch Status of Selected Counters
When bit 4 = 1, the status of the selected counters will be latched. When bit 4 = 0, the status will not be latched. The status byte format is described in Section 4.3.2, Interval Timer Status Byte Format Register.

Bit 3: Counter 2 Select
When bit 3 = 1, Counter 2 is selected for the latch command selected with bits 4 and 5. When bit 3 = 0, status and/or count will not be latched.

Bit 2: Counter 1 Select
When bit 2 = 1, Counter 1 is selected for the latch command selected with bits 4 and 5. When bit 2 = 0, status and/or count will not be latched.

Bit 1: Counter 0 Select
When bit 1 = 1, Counter 0 is selected for the latch command selected with bits 4 and 5. When bit 1 = 0, status and/or count will not be latched.

Bit 0: Reserved
Must be 0

4.3.1.2 Counter Latch Command

The Counter Latch Command latches the current count value at the time the command is received. This command is used to insure that the count read from the counter is accurate (particularly when reading a two-byte count). The count value is then read from each counter's count register (via the Counter Access Ports Register). One, two or all three counters may be latched with one Counter Latch Command.

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

The count must be read according to the programmed format. Specifically, if the Counter is programmed for two byte counts, two bytes must be read. The two bytes do not have to be read one right after the other (read, write, or programming operations for other counters may be inserted between the reads).

NOTES:

1. If a counter is programmed to read/write two-byte counts, a program must not transfer control between reading the first and second byte to another routine that also reads from that same counter. Otherwise, an incorrect count will be read. Finish reading the latched two-byte count before transferring control to another routine.
2. The Timer Counter Register bit definitions are different during the Counter Latch Command than for a normal Timer Counter Register write.

Bits[7:6]: Counter Selection
Bits 6 and 7 are used to select the counter for latching.

Bit 7	6	Function
0	0	latch counter 0 select
0	1	latch counter 1 select
1	0	latch counter 2 select
1	1	Read Back Command select

Bits[5:4]: Counter Latch Command
When bits[5:4] are both 0, the Counter Latch Command is selected during a write to the Timer Control Word Register. As noted above, the normal meanings (mode, countdown, r/w select) of the bits in the control register at I/O address 043h change when the Counter Latch Command is selected. Following the Counter Latch Command, I/O reads from the selected counter's I/O addresses produce the current latched count.

Bits[3:0]: Reserved
Must be 0.

4.3.2 INTERVAL TIMER STATUS BYTE FORMAT REGISTER

Address Offset:	Counter 0–040h Counter 1–041h Counter 2–042h
Default Value:	Bits[6:0] = X, Bit 7 = 0
Attribute:	Read Only
Size:	8 bits per counter

Each counter's status byte can be read following an Interval Timer Read Back Command. The Read Back Command is programmed through the Timer Control Word Register. If latch status is chosen (bit 4 = 0, Read Back Command) as a read back option for a given counter, the next read from the counter's Counter Access Ports Register returns the status byte. The status byte returns the countdown type, either BCD or binary; the counter operational mode; the read/write selection status; the Null count, also referred to as the count register status; and the current state of the counter OUT pin.

Bit 7: Counter OUT Pin State
When this bit is a 1, the OUT pin of the counter is also a 1. When this bit is a 0, the OUT pin of the counter is also a 0.

Bit 6: Count Register Status
Null Count, also referred to as the Count Status Register, indicates when the last count written to the Count Register (CR) has been loaded into the Counting Element (CE). The exact time this happens depends on the counter mode, but until the count is loaded into the counting element (CE), it can't be read from the counter. If the count is latched or read before the load time, the count value returned will not reflect the new count written to the register. When bit 6 = 0, the count has been transferred from CR to CE and is available for reading. When bit 6 = 1, the Null count condition exists. The count has not been transferred from CR to CE and is not yet available for reading.

Bits[5:4]: Read/Write Selection Status
Bits[5:4] reflect the read/write selection made through bits[5:4] of the control register. The binary codes returned during the status read match the codes used to program the counter read/write selection.

Bit 5	4	Function
0	0	Counter Latch Command
0	1	R/W Least Significant Byte (LSB)
1	0	R/W Most Significant Byte (MSB)
1	1	R/W LSB then MSB

Bits[3:1]: Mode Selection Status
Bits[3:1] return the counter mode programming. The binary code returned matches the code used to program the counter mode, as listed under the bit function above.

Bit 3	2	1	Mode Selected
0	0	0	0
0	0	1	1
X	1	0	2
X	1	1	3
1	0	0	4
1	0	1	5

Bit 0: Countdown Type Status
Bit reflects the current countdown type; either 0 for binary countdown or a 1 for binary coded decimal (BCD) countdown.

4.3.3 COUNTER ACCESS PORTS REGISTER

Address Offset:	Counter 0, System Timer-040h Counter 1, Refresh Request-041h Counter 2, Speaker Tone-042h
Default Value:	All bits undefined
Attribute:	Read/Write
Size:	8 bits per counter

Each of these I/O ports is used for writing count values to the Count Registers; reading the current count value from the counter by either an I/O read, after a counter-latch command, or after a Read Back Command; and reading the status byte following a Read Back Command.

Bits[7:0]: Counter Port Bit[x]
Each counter I/O port address is used to program the 16-bit Count Register. The order of programming, either LSB only, MSB only, or LSB then MSB, is defined with the Interval Counter Control Register at I/O port address 043h. The counter I/O port is also used to read the current count from the Count Register, and return the status of the counter programming following a Read Back Command.

4.3.4 BIOS TIMER REGISTER

Register Location:	Default = 78h–7Bh (Dword aligned)
Default Value:	0000xxxxh
Attribute:	Read/Write, Programmable
Size:	32 bit

A write to the BIOS Timer initiates a counting sequence. The timer can be initiated by writing either a 16-bit data portion or the entire 32-bit register (the upper 16 bits are don't cares). Bits[15:0] can be written with the initial count value to start the timer or read to check the current count value. It is the programmer's responsiblity to ensure that all 16 bits are written at the same time. After data is written into BIOS timer, the timer will start decrementing until it reaches zero. It will "freeze" at zero until the new count value is written.

The BIOS Timer consists of a single 32-bit register mapped in the I/O space on the location determined by the value written into the BIOS Timer Base Address Register. Bit 0 of the BIOS Timer Base Address Register enables/disables accesses to the BIOS Timer and must be 1 to enable access to the BIOS Timer Register. When the BIOS Timer is enabled, PCI accesses to the BIOS Timer Register do not flow through to the ISA Bus. If the BIOS Timer is disabled, accesses to the addresses assigned to the BIOS Timer Register flow through to the ISA bus. Note, however, that the counter continues to count normally.

Bits[31:16]: Reserved
Read as 0.

Bits[15:0]:
Timer count value.

4.4 Interrupt Controller Register Description

The SIO contains an ISA compatible interrupt controller that incorporates the functionality of two 82C59 interrupt controllers. The interrupt registers control the operation of the interrupt controller and can be accessed from the PCI Bus via PCI I/O space. In addition, some of the registers can be accessed from the ISA Bus via ISA I/O space. The bus access for each register is listed in Table 4.

4.4.1 ICW1—INITIALIZATION COMMAND WORD 1 REGISTER

Register Location:	INT CNTRL-1–020h INT CNTRL-2–0A0h
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bits per controller

A write to Initialization Command Word 1 starts the interrupt controller initialization sequence. Addresses 020h and 0A0h are referred to as the base addresses of CNTRL-1 and CNTRL-2, respectively.

An I/O write to the CNTRL-1 or CNTRL-2 base address with bit 4 equal to 1 is interpreted as ICW1. For SIO-based ISA systems, three I/O writes to "base address + 1" must follow the ICW1. The first write to "base address + 1" performs ICW2, the second write performs ICW3, and the third write performs ICW4.

ICW1 starts the initialization sequence during which the following automatically occur:

a. The edge sense circuit is reset. This means that following initialization, an interrupt request (IRQ) input must make a low-to-high transition to generate an interrupt.

b. The Interrupt Mask register is cleared.

c. IRQ7 input is assigned priority 7.

d. The slave mode address is set to 7.

e. Special Mask Mode is cleared and Status Read is set to IRR.

f. If IC4 was set to 0, then all functions selected by ICW4 are set to 0. However, ICW4 must be programmed in the SIO implementation of this interrupt controller, and IC4 must be set to a 1.

ICW1 has three significant functions within the SIO interrupt controller configuration. ICW4 is needed, so bit 0 must be programmed to a 1. There are two interrupt controllers in the system, so bit 1, SNGL, must be programmed to a 0 on both CNTRL-1 and CNTRL-2, to indicate a cascade configuration. LTIM, the interrupt controller IRQ edge/level detection control bit, defines the IRQ sensing mode for each controller. When bit 3 is a 0, each IRQ line on the selected controller is programmed for edge-triggered mode. This mode is signified by a low-to-high transition on an IRQ input line. When bit 3 is a 1, the controller is programmed in level-triggered mode, where a high level on an IRQ input indicates the presence of an interrupt request. LTIM is global for each controller. The incoming IRQs are either all edge-triggered or all level-triggered. Bit D4 must be a 1 when programming ICW1. OCW2 and OCW3 are also addressed at the same port as ICW1. This bit indicates that ICW1, and not OCW2 or OCW3, will be programmed during the write to this port.

Bit 2, ADI, and bits[7:5], A7-A5, are specific to an MSC-85 implementation. These bits are not used by the SIO interrupt controllers. Bits[7:5,2] should each be initialized to 0.

In the 82378ZB, bit 3, the LTIM bit, is not used by the interrupt controller and is always read as a 1.

Bits[7:5]: ICW/OCW Select
A7–A5 are MCS-85 implementation specific bits. They are not needed by the SIO. These bits should be 000 when programming the SIO.

Bit 4: ICW/OCW Select
Bit 4 must be a 1 to select ICW1. After the fixed initialization sequence to ICW1, ICW2, ICW3, and ICW4, the controller base address is used to write to OCW2 and OCW3. Bit 4 is a 0 on writes to these registers. A 1 on this bit at any time will force the interrupt controller to interpret the write as an ICW1. The controller will then expect to see ICW2, ICW3, and ICW4.

Bit 3: LTIM (Edge/Level Bank Select)
Ignored for the SIO.

Bit 2: ADI
Ignored for the SIO.

Bit 1: SNGL (Single or Cascade)
SNGL must be programmed to a 0 to indicate that two interrupt controllers are operating in cascade mode on the SIO.

Bit 0: IC4 (ICW4 Write Required)
This bit must be set to a 1. IC4 indicates that ICW4 needs to be programmed. The SIO requires that ICW4 be programmed to indicate that the controllers are operating in an 80x86 type system.

4.4.2 ICW2—INITIALIZATION COMMAND WORD 2 REGISTER

Address Offset:	INT CNTRL-1–021h INT CNTRL-2–0A1h
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bits per controller

ICW2 is used to initialize the interrupt controller with the five most significant bits of the interrupt vector address. The value programmed for bits[7:3] is used by the Host CPU to define the base address in the interrupt vector table for the interrupt routines associated with each IRQ on the controller. Typical ISA ICW2 values are 04h for CNTRL-1 and 70h for CNTRL-2.

Bits[7:3]: Interrupt Vector Base Address
Bits[7:3] define the base address in the interrupt vector table for the interrupt routines associated with each interrupt request level input. For CNTRL-1, a typical value is 00001, and for CNTRL-2, 10000.

The interrupt controller combines a binary code representing the interrupt level to receive service with this base address to form the interrupt vector that is driven out onto the bus. For example, the complete interrupt vector for IRQ[0] (CNTRL-1), would be 0000 1000b (CNTRL-1 [7:3] = 00001b and 000b representing IRQ[0]). This vector is used by the CPU to point to the address information that defines the start of the interrupt routine.

Bits[2:0]: Interrupt Request Level
When writing ICW2, these bits should all be 0. During an interrupt acknowledge cycle, these bits are programmed by the interrupt controller with the interrupt code representing the interrupt level to be serviced. This interrupt code is combined with bits[7:3] to form the complete interrupt vector driven onto the data bus during the second INTA# cycle. Section 5.0, Detailed Function Description, outlines each of these codes. The code is a simple three bit binary code: 000 represents IRQ0 (IRQ8), 001 IRQ1 (IRQ9), 010 IRQ2 (IRQ10), and so on until 111 IRQ7 (IRQ15).

4.4.3 ICW3—INITIALIZATION COMMAND WORD 3 REGISTER

Register Name:	Initialization Command Word 3 (Controller 1-Master Unit)
Address Offset:	INT CNTRL-1–021h
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bits

The meaning of ICW3 differs between CNTRL-1 and CNTRL-2. On CNTRL-1, the master controller, ICW3 indicates which CNTRL-1 IRQ line physically connects the INT output of CNTRL-2 to CNTRL-1. ICW3 must be programmed to 04h, indicating the cascade of the CNTRL-2 INT output to the IRQ[2] input of CNTRL-1.

An interrupt request on IRQ2 causes CNTRL-1 to enable CNTRL-2 to present the interrupt vector address during the second interrupt acknowledge cycle.

Bits[7:3]:
These bits must be programmed to zero.

Bit 2: Cascaded Interrupt Controller IRQ Connection
Bit 2 must always be programmed to a 1. This bit indicates that CNTRL-2, the slave controller, is cascaded on interrupt request line two (IRQ[2]). When an interrupt request is asserted to CNTRL-2, the IRQ goes through the priority resolver. After the slave

controller priority resolution is finished, the INT output of CNTRL-2 is asserted. However, this INT assertion does not go directly to the CPU. Instead, the INT assertion cascades into IRQ[2] on CNTRL-1. IRQ[2] must go through the priority resolution process on CNTRL-1. If it wins the priority resolution on CNTRL-1 and the CNTRL-1 INT signal is asserted to the CPU, the returning interrupt acknowledge cycle is really destined for CNTRL-2. The interrupt was originally requested at CNTRL-2, so the interrupt acknowledge is destined for CNTRL-2, and not a response for IRQ[2] on CNTRL-1.

When an interrupt request from IRQ[2] wins the priority arbitration, in reality an interrupt from CNTRL-2 has won the arbitration. Because bit 2 of ICW3 on the master is set to 1, the master knows which identification code to broadcast on the internal cascade lines, alerting the slave controller that it is responsible for driving the interrupt vector during the second INTA# pulse.

Bits[1:0]:
These bits must be programmed to zero.

4.4.4 ICW3—INITIALIZATION COMMAND WORD 3 REGISTER

Register Name:	Initialization Command Word 3 (Controller 2-Slave Unit)
Address Offset:	INT CNTRL-2–0A1h
Default Value:	All bits undefined
Attribute:	Write Only
Size:	8 bits

On CNTRL-2 (the slave controller), ICW3 is the slave identification code broadcast by CNTRL-1 from the trailing edge of the first INTA# pulse to the trailing edge of the second INTA# pulse. CNTRL-2 compares the value programmed in ICW3 with the incoming identification code. The code is broadcast over three SIO internal cascade lines. ICW3 must be programmed to 02h for CNTRL-2. When 010b is broadcast by CNTRL-1 during the INTA# sequence, CNTRL-2 assumes responsibility for broadcasting the interrupt vector during the second interrupt acknowledge cycle.

As an illustration, consider an interrupt request on IRQ[2] of CNTRL-1. By definition, a request on IRQ[2] must have been asserted by CNTRL-2. If IRQ[2] wins the priority resolution on CNTRL-1, the interrupt acknowledge cycle returned by the CPU following the interrupt is destined for CNTRL-2, not CNTRL-1. CNTRL-1 will see the INTA# signal, and knowing that the actual destination is CNTRL-2, will broadcast a slave identification code across the internal cascade lines. CNTRL-2 will compare this incoming value with the 010b stored in ICW3. Following a positive decode of the incoming message from CNTRL-1, CNTRL-2 will drive the appropriate interrupt vector onto the data bus during the second interrupt acknowledge cycle.

Bits[7:3]: Reserved
Must be 0.

Bits[2:0]: Slave Identification Code
The Slave Identification code must be programmed to 010b during the initialization sequence. The code stored in ICW3 is compared to the incoming slave identification code broadcast by the master controller during interrupt acknowledge cycles.

4.4.5 ICW4—INITIALIZATION COMMAND WORD 4 REGISTER

Address Offset:	INT CNTRL-1–021h INT CNTRL-2–0A1h
Default Value:	01h
Attribute:	Write Only
Size:	8 bits

Both SIO interrupt controllers must have ICW4 programmed as part of their initialization sequence. Minimally, the microprocessor mode bit, bit 0, must be set to a 1 to indicate to the controller that it is operating in an 80x86 based system. Failure to program this bit will result in improper controller operation during interrupt acknowledge cycles. Additionally, the Automatic End of Interrupt (AEOI) may be selected, as well as the Special Fully Nested Mode (SFNM) of operation.

The default programming for ICW4 is 01h, which selects 80x86 mode, normal EOI, buffered mode, and special fully nested mode disabled.

Bits 2 and 3 must be programmed to 0 for the SIO interrupt controller to function correctly.

Both bit 1, AEOI, and bit 4, SFNM, can be programmed if the system developer chooses to invoke either mode.

Bits[7:5]: Reserved
Must be 0.

Bit 4: SFNM (Special Fully Nested Mode)
Bit 4, SFNM, should normally be disabled by writing a 0 to this bit. If SFNM = 1, the special fully nested mode is programmed.

Bit 3: BUF (Buffered Mode)
Bit 3, BUF, must be programmed to 0 for the SIO. This is non-buffered mode. As illustrated above under bit functionality, different programming options are offered for bits 2 and 3. However, within the SIO interrupt unit, bits 2 and 3 must always be programmed to 00b.

Bit 2: Master/Slave in Buffered Mode
This bit is not used by the SIO interrupt unit. Bit 2 should always be programmed to 0.

Bit 1: AEOI (Automatic End of Interrupt)
This bit should normally be programmed to 0. This is the normal end of interrupt. If this bit is 1, the automatic end of interrupt mode is programmed.

Bit 0: Microprocessor Mode
The Microprocessor Mode bit must be programmed to 1 to indicate that the interrupt controller is operating in an 80x86-based system. Never program this bit to 0.

4.4.6 OCW1—OPERATIONAL CONTROL WORD 1 REGISTER

Address Offset:	INT CNTRL-1–021h INT CNTRL-2–0A1h
Default Value:	00h
Attribute:	Read/Write
Size:	8 bits

OCW1 sets and clears the mask bits in the Interrupt Mask Register (IMR). Each interrupt request line may be selectively masked or unmasked any time after initialization. A single byte is written to this register. Each bit position in the byte represents the same-numbered channel: bit 0 = IRQ[0], bit 1 = IRQ[1] and so on. Setting the bit to a 1 sets the mask, and clearing the bit to a 0 clears the mask. Note that masking IRQ[2] on CNTRL-1 will also mask all of controller 2's interrupt requests (IRQ8–IRQ15). Reading OCW1 returns the controller's mask register status.

The IMR stores the bits which mask the interrupt lines to be masked. The IMR operates on the IRR. Masking of a higher priority input will not affect the interrupt request lines of lower priority.

Unlike status reads of the ISR and IRR, for reading the IMR, no OCW3 is needed. The output data bus will contain the IMR whenever I/O read is active and the I/O port address is 021h or 0A1h (OCW1).

All writes to OCW1 must occur following the ICW1–ICW4 initialization sequence, since the same I/O ports are used for OCW1, ICW2, ICW3 and ICW4.

Bits[7:0]: Interrupt Request Mask (Mask [7:0])
When a 1 is written to any bit in this register, the corresponding IRQ[x] line is masked. For example, if bit 4 is set to a 1, then IRQ[4] will be masked. Interrupt requests on IRQ[4] will not set channel 4's interrupt request register (IRR) bit as long is the channel is masked.

When a 0 is written to any bit in this register, the corresponding IRQ[x] mask bit is cleared, and interrupt requests will again be accepted by the controller.

NOTE:
Masking IRQ[2] on CNTRL-1 will also mask the interrupt requests from CNTRL-2, which is physically cascaded to IRQ[2].

4.4.7 OCW2—OPERATIONAL CONTROL WORD 2 REGISTER

Address Offset: INT CNTRL-1-020h
INT CNTRL-2-0A0h
Default Value: Bit[4:0] = undefined,
Bit[7:5] = 001
Attribute: Write Only
Size: 8 bits

OCW2 controls both the Rotate Mode and the End of Interrupt Mode, and combinations of the two. The three high order bits in an OCW2 write represent the encoded command. The three low order bits are used to select individual interrupt channels during three of the seven commands. The three low order bits (labeled L2, L1 and L0) are used when bit 6 is set to a 1 during the command.

Following a PCIRST# and ICW initialization, the controller enters the fully nested mode of operation. Non-specific EOI without rotation is the default. Both rotation mode and specific EOI mode are disabled following initialization.

Bits[7:5]: Rotate and EOI Codes
R, SL, EOI-These three bits control the Rotate and End of Interrupt modes and combinations of the two. A chart of these combinations is listed above under the bit definition.

Bits 7	6	5	Function
0	0	1	Non-Specific EOI Command
0	1	1	Specific EOI Command
1	0	1	Rotate on Non-Specific EOI Command
1	0	0	Rotate in Auto EOI Mode (Set)
0	0	0	Rotate in Auto EOI Mode (Clear)
1	1	1	*Rotate on Specific EOI Command
1	1	0	*Set Priority Command
0	1	0	No Operation

NOTE:
* L0-L2 Are Used

Bits[4:3]: OCW2 Select
When selecting OCW2, bits 3 and 4 must both be 0. If bit 4 is a 1, the interrupt controller interprets the write to this port as an ICW1. Therefore, always ensure that these bits are both 0 when writing an OCW2.

Bits[2:0]: Interrupt Level Select (L2, L1, L0)
L2, L1, and L0 determine the interrupt level acted upon when the SL bit is active. A simple binary code, outlined above, selects the channel for the command to act upon. When the SL bit is inactive, these bits do not have a defined function; programming L2, L1 and L0 to 0 is sufficient in this case.

Bit 2	1	0	Interrupt Level
0	0	0	IRQ 0(8)
0	0	1	IRQ 1(9)
0	1	0	IRQ 2(10)
0	1	1	IRQ 3(11)
1	0	0	IRQ 4(12)
1	0	1	IRQ 5(13)
1	1	0	IRQ 6(14)
1	1	1	IRQ 7(15)

4.4.8 OCW3—OPERATIONAL CONTROL WORD 3 REGISTER

Address Offset: INT CNTRL-1-020h
INT CNTRL-2-0A0h
Default Value: Bit[6.0] = 0
Bit[7,4:2] = undefined,
Bit[5,1] = 1
Attribute: Read/Write
Size: 8 bits

OCW3 serves three important functions: Enable Special Mask Mode, Poll Mode control, and IRR/ISR register read control.

First, OCW3 is used to set or reset the Special Mask Mode (SMM). The Special Mask Mode can be used by an interrupt service routine to dynamically alter the system priority structure while the routine is executing, through selective enabling/disabling of the other channel's mask bits.

Second, the Poll Mode is enabled when a write to OCW3 is issued with bit 2 equal to 1. The next I/O read to the interrupt controller is treated like an interrupt acknowledge; a binary code representing the highest priority level interrupt request is released onto the bus.

Third, OCW3 provides control for reading the In-Service Register (ISR) and the Interrupt Request Register (IRR). Either the ISR or IRR is selected for reading with a write to OCW3. Bits 0 and 1 carry the encoded command to select either register. The next I/O read to the OCW3 port address will return the register status specified during the previous write. The register specified for a status read is retained by the interrupt controller. Therefore, a write to OCW3 prior to every status read command is unnecessary, provided the status read desired is from the register selected with the last OCW3 write.

Bit 7: Reserved
Must be 0.

Bit 6: SMM (Special Mask Mode)
If ESMM = 1 and SMM = 1 the interrupt controller enters Special Mask Mode. If ESMM = 1 and SMM = 0, the interrupt controller is in normal mask mode. When ESMM = 0, SMM has no effect.

Bit 5: ESMM (Enable Special Mask Mode)
When ESMM = 1, the SMM bit is enabled to set or reset the Special Mask Mode. When ESMM = 0, the SMM bit becomes a "don't care".

Bits[4:3]: OCW3 Select
When selecting OCW3, bit 3 must be a 1 and bit 4 must be 0. If bit 4 = 1, the interrupt controller interprets the write to this port as an ICW1. Therefore, always ensure that bits[4:3] = 01 when writing an OCW3.

Bit 2: Poll Mode Command
When bit 2 = 0, the Poll command is not issued. When bit 2 = 1, the next I/O read to the interrupt controller is treated as an interrupt acknowledge cycle. An encoded byte is driven onto the data bus representing the highest priority level requesting service.

Bits[1:0]: Register Read Command
Bits[1:0] provide control for reading the In-Service Register (ISR) and the Interrupt Request Register (IRR). When bit 1 = 0, bit 0 will not affect the register read selection. When bit 1 = 1, bit 0 selects the register status returned following an OCW3 read. If bit 0 = 0, the IRR will be read. If bit 0 = 1, the ISR will be read. Following ICW initialization, the default OCW3 port address read will be "read IRR". To retain the current selection (read ISR or read IRR), always write a 0 to bit 1 when programming this register. The selected register can be read repeatedly without reprogramming OCW3. To select a new status register, OCW3 must be reprogrammed prior to attempting the read.

Bit	1	0	Function
	0	0	No Action
	0	1	No Action
	1	0	Read IRQ Register
	1	1	Read IS Register

4.5 Control Registers

This section contains NMI registers, a real-time clock register, Port 92 Register, and the Digital Output Register.

4.5.1 NMISC—NMI STATUS AND CONTROL REGISTER

Address Offset: 061h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

This register is used to check the status of different system components, control the output of the speaker counter (Counter 2), and gate the counter output that drives the SPKR signal.

Bits 4, 5, 6, and 7 are read-only. When writing to this port, these bits must be written as 0's. Bit 6 returns the IOCHK# NMI status. This input signal comes from the ISA Bus. It is used for parity errors on memory cards plugged into the bus, and for other high priority interrupts. The current status of bit 3 enables or disables this NMI source. Bit 5 is the current state of the OUT pin of interval Timer 1, Counter 2. Bit 4 toggles from 1-0 or from 0-1 after every Refresh cycle. Following PCIRST#, bits 4 and 6 are both 0. Bit 5 is undetermined until Counter 2 is properly programmed. Bit 7 returns the PCI System Error status (SERR#). If 0, bit 7 indicates that SERR# was not pulsed active by a PCI agent. If 1, bit 7 indicates that SERR# was pulsed active by a PCI agent and that an NMI will be issued to the Host CPU. This NMI can be disabled with bit 2 of this register.

Bits 0–3 are both read and write. Bit 0 is the GATE input signal for Timer 1, Counter 2. The GATE input is used to disable counting in Counter 2. The Counter 2 output is ANDed with bit 1 to form the SPKR output signal. Bit 1 gates the Counter 2 OUT value. When bit 1 is disabled, the SPKR signal is disabled; when bit 1 is enabled, the SPKR output follows the value at the OUT pin of Counter 2. The Counter 2 OUT pin status can be checked by reading port 061h and checking bit 5. Bit 2 is used to enable the System Error (SERR#) signal. Bit 3 enables or disables the incoming IOCHK# NMI signal from the expansion bus. Each of these bits is reset to 0 following PCIRST#.

Bit 7: SERR# Status
Bit 7 is set if a system board agent (PCI devices or main memory) detects a system board error and pulses the PCI SERR# line. This interrupt is enabled by setting bit 2 to 0. To reset the interrupt, set bit 2 to 0 and then set it to 1. This bit is read-only. When writing to port 061h, bit 6 must be a 0.

Bit 6: IOCHK# NMI Source Status
Bit 6 is set if an expansion board asserts IOCHK# on the ISA/SIO bus. This interrupt is enabled by setting bit 3 to 0. To reset the interrupt, set bit 3 to 0 and then set it to 1. This bit is read-only. When writing to port 061h. bit 6 must be a 0

Bit 5: Timer Counter 2 OUT Status
The Counter 2 OUT signal state is reflected in bit 5. The value on this bit following a read is the current state of the Counter 2 OUT signal. Counter 2 must be programmed following a PCIRST# for this bit to have a determinate value. Bit 5 is read-only. When writing to port 061h, bit 5 must be a 0.

Bit 4: Refresh Cycle Toggle
The Refresh Cycle Toggle signal toggles from either 0 to 1 or 1 to 0 following every refresh cycle. This read-only bit is a 0 following PCIRST#. When writing to port 061h, bit 4 must be a 0.

Bit 3: IOCHK# NMI Enable
When bit 3 = 1, IOCHK# NMI's are disabled and cleared. When bit 3 = 0, IOCHK# NMI's are enabled. Following PCIRST#, bit 3 is reset to 0.

Bit 2: PCI SERR# Enable
When bit 2 = 1, the PCI System Error (SERR#) is disabled and cleared. When bit 2 = 0, SERR# is enabled. Following PCIRST#, bit 2 is a 0.

Bit 1: Speaker Data Enable
Speaker Data Enable is ANDed with the Counter 2 OUT signal to drive the SPKR output signal. When bit 1 = 0, the result of the AND is always 0 and the SPKR output is always 0. When bit 1 = 1, the SPKR output is equivalent to the Counter 2 OUT signal value. Following PCIRST#, bit 1 is a 0.

Bit 0: Timer Counter 2 Enable
When bit 0 = 0, Counter 2 counting is disabled. Counting is enabled when bit 0 = 1. This bit controls the GATE input to Counter 2. Following PCIRST#, the value of this bit is 0.

4.5.2 NMI ENABLE AND REAL-TIME CLOCK ADDRESS REGISTER

Address Offset: 070h
Default Value: Bit[6:0] = undefined, Bit 7 = 1
Attribute: Write Only
Size: 8 bits

The Mask register for the NMI interrupt is at I/O address 070h shown below. The most significant bit enables or disables all NMI sources including IOCHK# and the NMI Port. Write an 80h to port 70h to mask the NMI signal. This port is shared with the real-time clock. The real-time-clock uses the lower six bits of this port to address memory locations. Writing to port 70h sets both the enable/disable bit and the memory address pointer. Do not modify the contents of this register without considering the effects on the state of the other bits. Reads and writes to this register address flow through to the ISA Bus.

Bit 7: NMI Enable
Setting bit 7 to a 1 disables all NMI sources. Setting the bit to a 0 enables the NMI interrupt. Following PCIRST#, this bit is a 1.

Bits[6:0]: Real Time Clock Address
Used by the Real Time Clock on the Base I/O component to address memory locations. Not used for NMI enabling/disabling.

4.5.3 PORT 92 REGISTER

Address Offset: 92h
Default Value: 24h
Attribute: Read/Write
Size: 8 bits

This register is used to support the alternate reset (ALT__RST#) and alternate A20 (ALT__A20) functions. This register is only accessible if bit 6 in the Utility Bus Chip Select B Register is set to a 1. Reads and writes to this register location flow through to the ISA Bus.

Bits[7:6]: Reserved
Returns 00 when read.

Bit 5: Reserved
Returns a 1 when read.

Bit 4: Reserved
Returns a 0 when read.

Bit 3: Reserved
Returns a 0 when read.

Bit 2: Reserved
Returns a 1 when read.

Bit 1: ALT__A20 Signal Control
Writing a 0 to this bit causes the ALT__A20 signal to be driven low. Writing a 1 to this bit causes the ALT__A20 signal to be driven high.

Bit 0: Alternate System Reset
This read/write bit provides an alternate system reset function. This function provides an alternate means to reset the system CPU to effect a mode switch from Protected Virtual Address Mode to the Real Address Mode. This provides a faster means of reset than is provided by the Keyboard controller. This bit is set to a 0 by a system reset. Writing a 1 to this bit will cause the ALT__RST# signal to pulse active (low) for approximately 4 SYSCLK's. Before another ALT__RST# pulse can be generated, this bit must be written back to a 0.

4.5.4 DIGITAL OUTPUT REGISTER

Address Offset: 03F2h (Primary), 0372h (Secondary)
Default Value: Bit[7:4,2:0] = undefined, Bit 3 = 0
Attribute: Write only
Size: 8 bits

This register is used to prevent UBUSOE# from responding to DACK2# during a DMA read access to a floppy controller on the ISA Bus. If a second floppy (residing on the ISA Bus) is using DACK2# in conjunction with a floppy on the utility bus, this prevents the floppy on the utility bus and the utility bus transceiver from responding to an access targeted for the floppy on the ISA Bus. This register is also located in the floppy controller device. Reads and writes to this register location flow through to the ISA Bus.

Bits[7:4]: Not Used
These bits exist in the floppy controller.

Bit 3: DMA Enable
When this bit is a 1, the assertion of DACK# will result in UBUSOE# being asserted. If this bit is 0, DACK2# has no effect on UBUSOE#. This port bit also exists on the floppy controller. This bit defaults to disable (0).

Bits[2:0]: Not Used
These bits exist in the floppy controller.

4.5.5 RESET UBUS IRQ12 REGISTER

Address Offset: 60h
Default Value: N/A
Attribute: Read only
Size: 8 bits

This address location (60h) is used to clear the mouse interrupt function to the CPU. Reads to this address are monitored by the SIO. When the mouse interrupt function is enabled (bit 4 of the ISA Clock Divisor Register is 1), the mouse interrupt function is provided on the IRQ12/M input signal. In this mode, a mouse interrupt generates an interrupt through IRQ13 to the Host CPU. A read of 60h releases IRQ12. If bit 4 = 0 in the ISA Clock Divisor Register, a read of address 60h has no effect on IRQ12/M. Reads and writes to this register flow through to the ISA Bus. For additional information, see the IRQ12/M description in Section 3.0, Signal Description.

Bits[7:0]: Reset IRQ12
No specific pattern. A read of address 60h executes the command.

4.5.6 COPROCESSOR ERROR REGISTER

Address Offset: F0h
Default Value: N/A
Attribute: Write only
Size: 8 bits

This address location (F0h) is used when the SIO is programmed for coprocessor error reporting (bit 5 of the ISA Clock Divisor Register is 1). Writes to this address are monitored by the SIO. In this mode, the SIO generates an interrupt (INT) to the CPU when it receives an error signal (FERR# asserted) from the CPU's coprocessor. Writing address F0h, when FERR# is asserted, causes the SIO to assert IGNNE# and negate IRQ13. IGNNE# remains asserted until FERR# is negated. If FERR# is not asserted, writing to address F0h does not effect IGNNE#. Reads and writes to this register flow through to the ISA Bus. For additional information, see the IGNNE# description in Section 3.0, Signal Description.

Bits[7:0]: Reset IRQ12
No specific pattern. A write to address F0h executes the command.

4.5.7 ELCR—EDGE/LEVEL CONTROL REGISTER

Address Offset: INT CNTRL-1-04D0h
INT CNTRL-2-04D1h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

The Edge/Level Control Register is used to set the interrupts to be triggered by either the signal edge or the logic level. INT0, INT1, INT2, INT8, INT13 must be set to edge sensitive. After a reset, all the INT signals are set to edge sensitive. Figure 4 shows which bit numbers represent the various INT signals.

Each IRQ to which a PCI interrupt is steered into (see the PIRQ Route Control Register) must have its interrupt set to level sensitive.

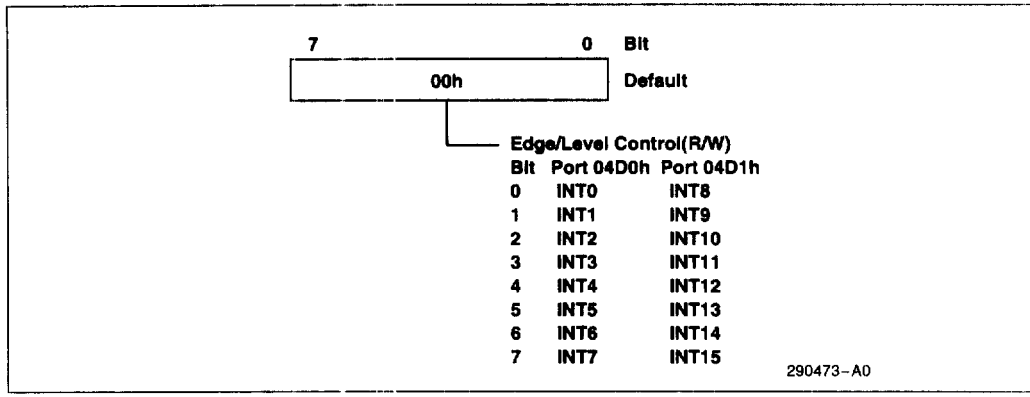


Figure 4. Edge/Level Select Register

Register Location: 04D0h-INT CNTRL-1
04D1h-INT CNTRL-2

04D0h-INT CNTRL-1 Register

Bit[7:0]: Edge/Level Select
These bits select if the interrupts are triggered by either the signal edge or the logic level. A 0 bit represents an edge sensitive interrupt, and a 1 is for level sensitive. The following bits MUST be set to 0:

Port 04D0h (INT-CNTRL-1)

0-INT0	0	Reserved. Read as zero.
1-INT1	0	Reserved. Read as zero.
2-INT2	0	Reserved. Read as zero.
3-INT3	x	
4-INT4	x	
5-INT5	x	
6-INT6	x	
7-INT7	x	

x = selectable to either a 0 or a 1,
0 = edge sensitive, 1 = level sensitive

After reset, this register is set to 00h.

04D1h-INT CNTRL-2 Register

Bit[7:0]: Edge/Level Select
These bits select if the interrupts are triggered by either the signal edge or the logic level. A 0 bit represents an edge sensitive interrupt, and a 1 is for level sensitive. The following bits MUST be set to 0:

Port 04D1h (INT-CNTRL-2)

0-INT8	0	Reserved. Read as zero.
1-INT9	x	
2-INT10	x	
3-INT11	x	
4-INT12	x	
5-INT13	0	Reserved. Read as zero.
6-INT14	x	
7-INT15	x	

x = selectable to either a 0 or a 1,
0 = edge sensitive, 1 = level sensitive

After reset, this register is set to 00h.

4.6. Power Management Registers

This section contains the Power Management Registers located in non-configuration space.

4.6.1 APMC—ADVANCED POWER MANAGEMENT CONTROL PORT

Address Offset: 0B2h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

Bits[7:0]: APMC[7:0]
APM Control Port. Readable/writeable at system I/O address 0B2h. Used to pass an APM command between the OS and the SMI handler. Writes to this port not only store data in the APMC register, but also generate an SMI when the SAPMCEN bit is set. Reads to this port will not generate an SMI. If CSTPCLKEN is set, a read from the APMC will cause STPCLK# to be asserted.

4.6.2 APMS—ADVANCED POWER MANAGEMENT STATUS PORT

Address Offset: 0B3h
Default Value: 00h
Attribute: Read/Write
Size: 8 bits

Bits[7:0]: APMS[7:0]
Readable/writeable at system address 0B3h. Used to pass data between the OS and the SMI handler.

5.0 DETAILED FUNCTIONAL DESCRIPTION

5.1 PCI Interface

5.1.1 PCI COMMAND SET

Bus commands indicate to the slave the type of transaction the master is requesting. Bus Commands are encoded on the C/BE[3 0]# lines during the address phase of a PCI cycle.

5.1.2 PCI BUS TRANSFER BASICS

Details of PCI Bus operations can be found in the *Peripheral Component Interconnect (PCI) Specification*. Only details of the PCI Bus unique to the SIO are included in this data sheet.

Table 7. PCI Commands

C/BE[3:0]#	Command Type As Slave	Supported As Slave	Supported As Master
0000	Interrupt Acknowledge	Yes	No
0001	Special Cycle(4)	No/Yes	No
0010	I/O Read	Yes	No
0011	I/O Write	Yes	No
0100	Reserved(3)	No	No
0101	Reserved(3)	No	No
0110	Memory Read	Yes	Yes
0111	Memory Write	Yes	Yes
1000	Reserved(3)	No	No
1001	Reserved(3)	No	No
1010	Configuration Read	Yes	No
1011	Configuration Write	Yes	No
1100	Memory Read Multiple	No(2)	No
1101	Reserved(3)	No	No
1110	Memory Read Line	No(2)	No
1111	Memory Write and Invalidate	No(1)	No

NOTES:
1. Treated as Memory Write.
2. Treated as Memory Read.
3. Reserved Cycles are considered invalid by the SIO and are to be competely ignored. All internal address decoding is ignored and DEVSEL# is never to be asserted.
4. The 82378 responds to a Stop Grant Special Cycle.

5.1.2.1 PCI Addressing

PCI address decoding uses the AD[31:0] signals. AD[31:2] are always used for address decoding while the information contained in the two low order bits AD[1:0] varies for memory, I/O, and configuration cycles.

For I/O cycles, AD[31:0] are decoded to provide a byte address. AD[1:0] are used for generation of DEVSEL# only and indicate the least significant valid byte involved in the transfer. For example, if only BE0# is asserted, AD[1:0] are 00. If only BE3# is asserted, then AD[1:0] are 11. If BE3# and BE2# are asserted, AD[1:0] are 10. If all BEx#'s are asserted, then AD[1:0] are 00. The byte enables determine which byte lanes contain valid data. The SIO requires that PCI accesses to byte-wide internal registers must assert only one byte enable.

When the SIO is the target of any PCI transaction in which BE[3:0]# = 1111, the SIO terminates the cycle normally by asserting TRDY#. No data is written into the SIO during write cycles and the data driven by the SIO during read cycles is indeterminate.

For memory cycles, accesses are decoded as Dword accesses. This means that AD[1:0] are ignored for decoding memory cycles. The byte enables determine which byte lanes contain valid data. When the SIO is a PCI master, it drives 00 on AD[1:0] for all memory cycles.

For configuration cycles, DEVSEL# is a function of IDSEL and AD[1:0]. DEVSEL# is selected during a configuration cycle only if IDSEL is active and both AD[1:0] = 00. The cycle is ignored by the SIO if either AD1 or AD0 is non-zero. Configuration registers are selected as Dwords using AD[7:2]. The byte enables determine which byte lanes contain valid data.

5.1.2.2 DEVSEL# Generation

As a PCI slave, the SIO asserts the DEVSEL# signal to indicate it is the slave of the PCI transaction. DEVSEL# is asserted when the SIO positively or subtractively decodes the PCI transaction. The SIO asserts DEVSEL# (claim the transaction) before it issues any other slave response, i.e., TRDY#, STOP#, etc. After the SIO asserts DEVSEL#, it does not negate DEVSEL# until the same edge that the master uses to negate the final IRDY#.

It is expected that most (perhaps all) PCI target devices will be able to complete a decode and assert DEVSEL# within 1 or 2 clocks of FRAME#. Since the SIO subtractively decodes all unclaimed PCI cycles (except configuration cycles), it provides a configuration option to pull in (by 1 or 2 clocks) the edge when the SIO samples DEVSEL#. This allows faster access to the expansion bus. Use of such an option is limited by the slowest positive decode agent on the bus. This is described in more detail in Section 5.5.1.4, Subtractively Decoded Cycles to ISA.

As a PCI master, the SIO waits for 5 PCICLKs after the assertion of FRAME# for a slave to assert DEVSEL#. If the SIO does not receive DEVSEL# in this time, it will master-abort the cycle. See Section 5.1.3.1, SIO as MasterMaster-Initiated Termination, for further details.

5.1.2.3 Basic PCI Read Cycles (I/O and Memory)

As a PCI master, the SIO only performs memory read transfers (i.e. I/O read transfers are not supported). When reading data from PCI memory, the SIO requests a maximum of 8 bytes via a two data phase burst read cycle to fill its internal 8 byte line buffer. If the line buffer is programmed for single transaction mode, fewer bytes are requested (refer to Section 5.6.1, DMA/ISA Master Line Buffer). Read cycles from PCI memory are generated on behalf of ISA masters and DMA devices.

As a PCI slave, the SIO responds to both I/O read and memory read transfers. For multiple read transactions, the SIO always target-terminates after the first data read transaction by asserting STOP# and TRDY# at the end of the first data phase. For single read transactions, the SIO finishes the cycle in a normal fashion, by asserting TRDY# without asserting STOP#.

5.1.2.4 Basic PCI Write Cycles (I/O and Memory)

As a PCI master, the SIO generates a PCI memory write cycle when it decodes an ISA-originated/PCI-bound memory write cycle. I/O write cycles are never initiated by the SIO. When writing data to PCI memory, the SIO writes a maximum of 4 bytes via a single data transaction write cycle. If the SIO's internal ISA master/DMA line buffer is programmed for single transaction mode, fewer bytes will be generated (refer to Section 5.6.1, DMA/ISA Master Line Buffer). In either case, only one data transaction will be performed. Cycles to PCI memory are generated on behalf of ISA masters, DMA devices, and the SIO when the SIO needs to flush the ISA master/DMA line buffer.

As a PCI master, the SIO drives the AD0 and AD1 signals low during the address phase of the cycle. This is done to indicate to the slave that the address will increment during the transfer. If there is no response on the PCI Bus, the SIO will master-abort due to the DEVSEL# time out.

As a PCI slave, the SIO will respond to both I/O write and memory write transfers. For multiple write transactions, the SIO will always target-terminate after the first data write transaction by asserting STOP# and TRDY# at the end of the first data phase. For single write transactions, the SIO will finish the cycle normally by asserting TRDY# without asserting STOP#.

5.1.2.5 Configuration Cycles

The configuration read or write command defined by the bus control signals C/BE[3:0]# is used to configure the SIO. During the address phase of the configuration read or write command, the SIO will sample its IDSEL (ID select). If IDSEL is active and AD[1:0] are both zero, the SIO generates DEVSEL#. Otherwise, the cycle is ignored by the SIO. During the configuration cycle address phase, bits AD[7:2] and C/BE[3:0]# are used to select particular bytes within a configuration register. Note that IDSEL is normally a "don't care" except during the address phase of a transaction.

NOTE:
An unclaimed configuration cycle is never forwarded to the ISA Bus.

5.1.2.6 Interrupt Acknowledge Cycle

The interrupt acknowledge command is a single byte read implicitly addressed to the SIO's interrupt controller. The address bits are logical "don't cares" during the address phase and the byte enables will indicate to the SIO an 8-bit interrupt vector is to be returned on AD[7:0]. The SIO converts this single cycle transfer into two cycles that the internal 8259 pair can respond to (see Section 5.8, Interrupt Controller). The SIO will hold the PCI Bus in wait states until the 8 bit interrupt vector is returned.

SIO responses to an interrupt acknowledge cycle can be disabled by setting bit 5 in the PCI Control Register to a 0. However, if disabled, the SIO will still respond to accesses to the interrupt register set and allow poll mode functions.

5.1.2.7 Exclusive Access

The SIO marks itself locked anytime it is the slave of the access and LOCK# is sampled negated during the address phase. As a locked slave, the SIO responds to a master only when it samples LOCK# negated and FRAME# asserted. The locking master may negate LOCK# at the end of the last data phase. The SIO unlocks itself when FRAME# and LOCK# are both negated. The SIO will respond by asserting STOP# with TRDY# negated (retry) to all transactions when LOCK# is asserted during the address phase.

Locked cycles are never generated by the SIO.

5.1.2.8 PCI Special Cycle

When the SCE bit (bit 3) in the COM PCI configuration register (configuration offset 04h) is set to a "0", the SIO will ignore all PCI Special Cycles. When the SCE bit is set to a "1", the SIO will recognize PCI Special Cycles.

The only PCI Special Cycle currently recognized is the Stop Grant Special Cycle which is broadcast onto the PCI bus when an S-series processor enters the Stop Grant State. The SCE bit must be set to a "1" when the Stop Clock feature is being used.

5.1.3 TRANSACTION TERMINATION

The SIO supports both Master-initiated Termination as well as Target-initiated Termination.

5.1.3.1 SIO As Master—Master-Initiated Termination

The SIO supports two forms of master-initiated termination:

1. Normal termination of a completed transaction.
2. Abnormal termination due to no slave responding to the transaction (Abort).

Figure 5 shows the SIO performing master-abort termination. This occurs when no slave responds to the SIO's master transaction by asserting DEVSEL# within 5 PCICLK's after FRAME# assertion. This master-abort condition is abnormal and it indicates an error condition. The SIO will not retry the cycle. The Received Master-abort Status bit in the PCI Status Register will be set indicating that the SIO experienced a master-abort condition.

If an ISA master or the DMA is waiting for the PCI cycle to terminate (CHRDY negated), the master-abort condition will cause the SIO to assert CHRDY to terminate the ISA cycle. Note that write data will be lost and the read data will be all 1's at the end of the cycle. This is identical to the way an unclaimed cycle is handled on the "normally ready" ISA Bus. If the line buffer is the requester of the PCI transaction, the master-abort mechanism will end the PCI cycle, but no data will be transferred into or out of the line buffer. The line buffer will not be allowed to retry the cycle.

5.1.3.2 SIO As A Master—Response To Target-Initiated Termination

SIO's response as a master to target-termination:

1. For a target-abort, the SIO will not retry the cycle. If an ISA master or the DMA is waiting for the PCI cycle to complete (CHRDY negated), the target-abort condition will cause the SIO to assert CHRDY and end the cycle on the ISA Bus. If the ISA master or DMA device was reading from PCI memory, the SIO will drive all 1's on the data lines of the ISA Bus. The Received Target-abort Status bit in the PCI Status Register will be set indicating that the SIO experienced a target-abort condition.
2. If the SIO is retried as a master on the PCI Bus, it will remove it's request for 2 PCI clocks before asserting it again to retry the cycle.
3. If the SIO is disconnected as a master on the PCI Bus, it will respond very much as if it had been retried. The difference between retry and disconnect is that the SIO did not see any data phase for the retry. Disconnect may be generated by a PCI slave when the SIO is running a burst memory read cycle to fill it's 8-byte Line Buffer. In this case, the SIO may need to finish a multi-data phase transfer, and thus, must recycle through arbitration as required for a retry. An example of this is when the on-board DMA requests an 8-byte Line Buffer transfer and the SIO is disconnected before the Line Buffer is completely filled.

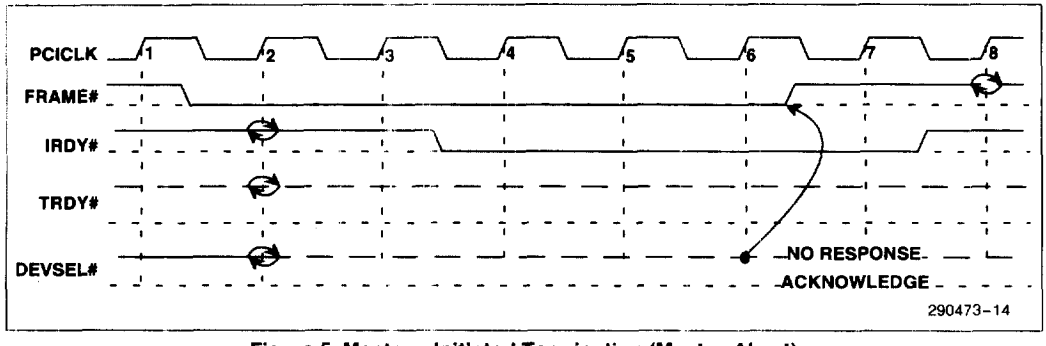


Figure 5. Master—Initiated Termination (Master-Abort)

5.1.3.3 SIO As A Target—Target-Initiated Termination

The SIO supports three forms of Target-initiated Termination:

Disconnect	Disconnect refers to termination requested because the SIO is unable to respond within the latency guidelines of the PCI specification. Note that this is not usually done on the first data phase.
Retry	Retry refers to termination requested because the target is currently in a state which makes it unable to process the transaction.
Abort	Abort refers to termination requested because the target will never be able to respond to the transaction.

The SIO will initiate Disconnect for PCI-originated/ISA-bound cycles after the first data phase due to incremental latency requirements. Since the SIO has only one Posted Write Buffer and every PCI to ISA incremental data phase will take longer than the specified 8 clocks, the SIO will always terminate burst cycles with a disconnect protocol. An example of this is when the SIO receives a burst memory write. Since the SIO only has one Posted Write Buffer, the transaction will automatically be disconnected after the first data phase.

The SIO will retry PCI masters:

1. For memory write cycles when the posted write buffer is full.
2. When the pending PCI cycle initiates some type of buffer management activity.
3. When the SIO is locked as a resource and a PCI master tries to access the SIO without negating the LOCK# signal in the address phase.
4. When the ISA Bus is occupied by an ISA master or DMA.

Target-abort is issued by the SIO when the internal SIO registers are the target of a PCI master I/O cycle and more than one byte enable is active. Accesses to the BIOS Timer Register and the Scatter/Gather Descriptor Table Pointer Registers are exceptions to this rule. Accesses to the Scatter/Gather Descriptor Table Pointer Register must be 32-bits wide and accesses to the BIOS Timer Register must be 16- or 32-bits wide. These accesses will not result in a SIO target-abort. The SIO responds with a target-abort since the registers must be accessed as 8-bit quantities. Target-abort resembles a retry, although the SIO also negates DEVSEL# along with the assertion of STOP#. Bit 11 in the Device Status Register is set to a 1 when the SIO target-aborts.

5.1.4 BUS LATENCY TIME-OUT

5.1.4.1 Master Latency Timer

Because the SIO only bursts a maximum of two Dwords, the PCI master latency timer is not implemented.

5.1.4.2 Target Incremental Latency Mechanism

As a slave, the SIO supports the Incremental Latency Mechanism for PCI to ISA cycles. The PCI specification states that for multi-data phase PCI cycles, if the incremental latency from current data phase (N) to the next data phase (N+1) is greater than 8 PCICLK's, then the slave must manipulate TRDY# and STOP# to stop the transaction upon completion of the current data phase (N). Since all PCI-originated (SIO is a slave)/ISA-bound cycles will require greater than the stated 8 PCICLK's, the SIO will automatically terminate these cycles after the first data phase. Note that latency to the first data phase is not restricted by this mechanism.

5.1.5 PARITY SUPPORT

As a master, the SIO generates address parity for read and write cycles, and data parity for write cycles. As a slave, the SIO generates data parity for read cycles. The SIO does not check parity and does not generate SERR#.

PAR is the calculated parity signal. PAR is "even" parity and is calculated on 36 bits; the 32 AD[31:0] signals plus the 4 C/BE[3:0]# signals. "Even" parity means that the number of 1's within the 36 bits plus PAR are counted and the sum is always even. PAR is always calculated on 36 bits, regardless of the valid byte enables. PAR is only guaranteed to be valid one PCI clock after the corresponding address or data phase.

5.1.6 RESET SUPPORT

The PCIRST# pin acts as the SIO hardware reset pin.

During Reset

AD[31:0], C/BE[3:0]#, and PAR are always driven low by the SIO from the leading edge of PCIRST#. FRAME#, IRDY#, TRDY#, STOP#, DEVSEL#, MEMREQ#, FLSHREQ#, CPUGNT#, GNT0#/ SIOREQ#, and GNT1#/ RESUME# are tri-stated from the leading edge of PCIRST#.

GNT2# and GNT3# are tri-stated from the leading edge of PCIRST#.

After Reset

AD[31:0], C/BE[3:0]#, and PAR are always tri-stated from the trailing edge of PCIRST#. If the internal arbiter is enabled (CPUREQ# sampled high on the trailing edge of PCIRST#), the SIO will drive these signals low again (synchronously 2-5 PCICLKs later) until the bus is given to another master. If the internal arbiter is disabled (CPUREQ# sampled low on the trailing edge of PCIRST#), these signals remain tri-stated until the SIO is required to drive them valid as a master or slave.

FRAME#, IRDY#, TRDY#, STOP#, and DEVSEL# remain tri-stated until driven by the SIO as either a master or a slave. MEMREQ#, FLSHREQ#, CPUGNT#, GNT0#/SIOREQ#, and GNT1#/RESUME# are tri-stated until driven by the SIO.

GNT2# and GNT3# are tri-stated until driven by the SIO.

After PCIRST, MEMREQ# and FLSHREQ# are driven inactive asynchronously from PCIRST# inactive. CPUGNT#, GNT0#/SIOREQ#, and GNT1#/ RESUME# are driven based on the arbitration scheme and the asserted REQx#'s.

GNT2# and GNT3# are also driven based on the arbitration scheme and the asserted REQx#'s.

5.1.7 DATA STEERING

Data steering logic internal to the SIO provides the assembly/disassembly, copy up/copy down mechanism for cycles between the 32-bit PCI data bus and the 16-bit ISA Bus. The steering logic ensures that the correct bytes are steered to the correct byte lane and that multiple cycles are run where applicable.

5.2 PCI Arbitration Controller

The 82378 contains a PCI Bus arbiter that supports six PCI masters; the Host Bridge, SIO, and four other masters. The SIO's REQ#/GNT# lines are internal. The integrated arbiter can be disabled by asserting CPUREQ# during PCIRST# (see Section 5.2.7, Power-up Configuration). When disabled, the SIO's REQ#, GNT#, and RESUME# signals become visible for an external arbiter. The internal arbiter is enabled upon power-up.

The internal arbiter contains several features that contribute to system efficiency:

- Use of a RESUME# signal to re-enable a backed-off initiator in order to minimize PCI Bus thrashing when the SIO generates a retry (Section 5.2.4.1).
- A programmable timer to re-enable retried initiators after a programmable number of PCICLK's (Section 5.2.4.2).
- The CPU (host bridge) can be optionally parked on the PCI Bus (Section 5.2.5).
- A programmable PCI Bus lock or PCI resource lock function (Section 5.2.6).

The PCI arbiter is also responsible for control of the Guaranteed Access Time (GAT) mode signals (Section 5.2.3.2).

5.2.1 ARBITRATION SIGNAL PROTOCOL

The internal arbiter follows the PCI arbitration method as outlined in the *Peripheral Component Interconnect (PCI) Specification*. The SIO's arbiter is discussed in this section.

5.2.1.1 Back-To-Back Transactions

The SIO as a master does not generate fast back-to-back accesses since it does not know if it is accessing the same target.

The SIO as a target supports fast back-to-back transactions. Note that for back-to-back cycles, the SIO treats positively decoded accesses and subtractively decoded accesses as different targets. Therefore, masters can only run fast back-to-back cycles to positively decoded addresses or to subtractively decoded addresses. Fast back-to-back cycles must not mix positive and subtractive decoded addresses. See the address decoding section to determine what addresses the SIO positively decodes and subtractively decodes.

5.2.2 PRIORITY SCHEME

The PCI arbitration priority scheme is programmable through the PCI Arbiter Priority Control and Arbiter Priority Control Extension Register. The arbiter consists of four banks that can be configured for the six masters to be arranged in a purely rotating priority scheme, one of twenty-four fixed priority schemes, or a hybrid combination (Figure 6).

Note that SIOREQ#/SIOGNT# are SIO internal signals.

The PCI Arbiter Priority Control (PAPC) and PCI Arbiter Priority Control Extension Register bits are shown below:

PCI Arbiter Priority Control Register Bits (PAPC)

Bit	Description
7	Bank 3 Rotate Control
6	Bank 2 Rotate Control
5	Bank 1 Rotate Control
4	Bank 0 Rotate Control
3	Bank 2 Fixed Priority Mode select B
2	Bank 2 Fixed Priority Mode select A
1	Bank 1 Fixed Priority Mode select
0	Bank 0 Fixed Priority Mode select

PCI Arbiter Priority Control Extension Register Bits (ARBPRIX)

Bit	Description
7:1	Reserved. Read as 0
0	Bank 3 Fixed Priority Mode select

PAPC defaults to 04h and ARBPRIX to 00h at reset selecting fixed mode #10 (Table 8) with the CPU the highest priority device guaranteeing access to BIOS.

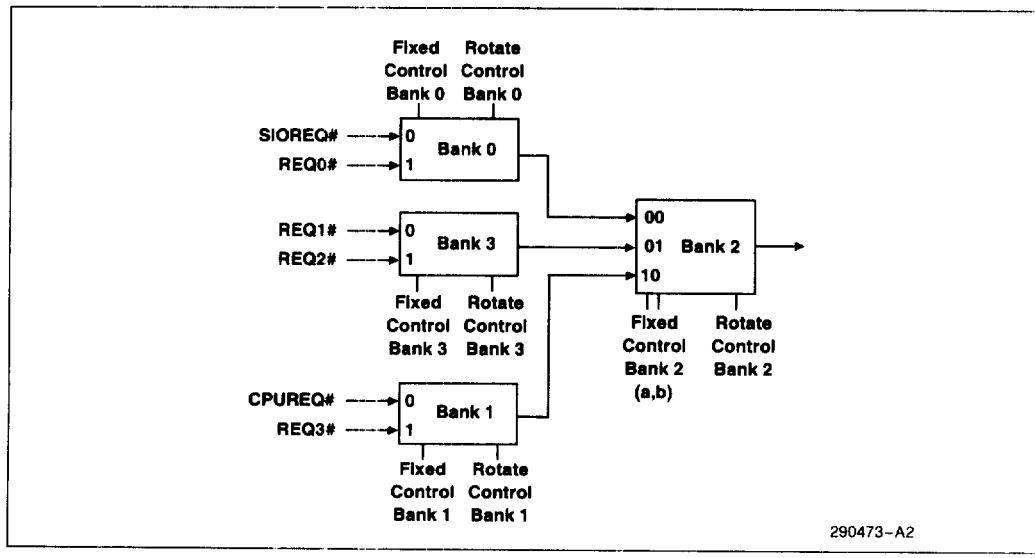


Figure 6. Arbiter Configuration Diagram for 82378ZB

5.2.2.1 Fixed Priority Mode

The 24 selectable fixed priority schemes are listed in Table 8.

Table 8. Fixed Priority Mode Bank Control Bits

Mode	Bank					Priority					
	3	2b	2a	1	0	Highest					Lowest
00	0	0	0	0	0	SIOREQ#	REQ0#	REQ2#	REQ3#	CPUREQ#	REQ1#
01	0	0	0	0	1	REQ0#	SIOREQ#	REQ2#	REQ3#	CPUREQ#	REQ#
02	0	0	0	1	0	SIOREQ#	REQ0#	REQ2#	REQ3#	REQ1#	CPUREQ#
03	0	0	0	1	1	REQ0#	SIOREQ#	REQ2#	REQ3#	REQ1#	CPUREQ#
04	0	0	1	0	0	CPUREQ#	REQ1#	SIOREQ#	REQ0#	REQ2#	REQ3#
05	0	0	1	0	1	CPUREQ#	REQ1#	REQ0#	SIOREQ#	REQ2#	REQ3#
06	0	0	1	1	0	REQ1#	CPUREQ#	SIOREQ#	REQ0#	REQ2#	REQ3#
07	0	0	1	1	1	REQ1#	CPUREQ#	REQ0#	SIOREQ#	REQ2#	REQ3#
08	0	1	0	0	0	REQ2#	REQ3#	CPUREQ#	REQ1#	SIOREQ#	REQ0#
09	0	1	0	0	1	REQ2#	REQ3#	CPUREQ#	REQ1#	REQ0#	SIOREQ#
0A	0	1	0	1	0	REQ2#	REQ3#	REQ1#	CPUREQ#	SIOREQ#	REQ0#
0B	0	1	0	1	1	REQ2#	REQ3#	REQ1#	CPUREQ#	REQ0#	SIOREQ#
0C–0F	0	1	1	x	x	Reserved					
10	1	0	0	0	0	SIOREQ#	REQ0#	REQ3#	REQ2#	CPUREQ#	REQ1#
11	1	0	0	0	1	REQ0#	SIOREQ#	REQ3#	REQ2#	CPUREQ#	REQ1#
12	1	0	0	1	0	SIOREQ#	REQ0#	REQ3#	REQ2#	REQ1#	CPUREQ#
13	1	0	0	1	1	REQ0#	SIOREQ#	REQ3#	REQ2#	REQ1#	CPUREQ#
14	1	0	1	0	0	CPUREQ#	REQ1#	SIOREQ#	REQ0#	REQ3#	REQ2#
15	1	0	1	0	1	CPUREQ#	REQ1#	REQ0#	SIOREQ#	REQ3#	REQ2#
16	1	0	1	1	0	REQ1#	SPUREQ#	SIOREQ#	REQ0#	REQ3#	REQ2#
17	1	0	1	1	1	REQ1#	CPUREQ#	REQ0#	SIOREQ#	REQ3#	REQ2#
18	1	1	0	0	0	REQ3#	REQ2#	CPUREQ#	REQ1#	SIOREQ#	REQ0#
19	1	1	0	0	1	REQ3#	REQ2#	CPUREQ#	REQ1#	REQ0#	SIOREQ#
1A	1	1	0	1	0	REQ3#	REQ2#	REQ1#	CPUREQ#	SIOREQ#	REQ0#
1B		1	0	1	1	REQ3#	REQ2#	REQ1#	CPUREQ#	REQ0#	SIOREQ#
1C–1F	1	1	1	x	x	Reserved					

The fixed bank control bit(s) selects which requester is the highest priority device within that particular bank. For fixed priority mode, bits[7:4] of the PAPC Register and bit zero of ARBPRIX must be 0's (rotate mode disabled).

The selectable fixed priority schemes provide 24 of the 64 possible fixed mode permutations possible for the six masters.

5.2.2.2 Rotating Priority Mode

When any bank rotate control bit is set to a one, that particular bank rotates between the requesting inputs. Any or all banks can be set in rotate mode. If all four banks are set in rotate mode, the six supported masters are all rotated and the arbiter is in a pure rotating priority mode. If, within a rotating bank, the highest priority device (a) does not have an active request, the lower priority device (b or c) will be granted the bus. However, this does not change the rotation scheme. When the bank toggles, device b is the highest priority. Because of this, the maximum latency a device can encounter is two complete rotations.

5.2.2.3 Mixed Priority Mode

Any combination of fixed priority and rotate priority modes can be used in different arbitration banks to achieve a specific arbitration scheme.

5.2.2.4 Locking Masters

When a master acquires the LOCK# signal, the arbiter gives that master highest priority until the LOCK# signal is negated and FRAME# is negated. This ensures that a master that locked a resource will eventually be able to unlock that same resource.

5.2.3 MEMREQ#, FLSHREQ#, AND MEMACK# PROTOCOL

Before an ISA master or the DMA can be granted the PCI Bus, it is necessary that all PCI system posted write buffers be flushed (including the SIO's Posted Write Buffer). Also, since the ISA originated cycle could access memory on the host bridge, it's possible that the ISA master or the DMA could be held in wait states (via IOCHRDY) waiting for the host bridge arbitration for longer than the 2.5 μs ISA specification. The SIO has an optional mode called the Guaranteed Access Time Mode (GAT) that ensures that this timing specification is not violated. This is accomplished by delaying the ISA REQ# signal to the requesting master or DMA until the ISA Bus, PCI Bus, and the System Memory Bus are arbitrated for and owned.

Three PCI sideband signals, MEMREQ#, FLSHREQ#, and MEMACK# are used to support the System Posted Write Buffer Flushing and Guaranteed Access Time mechanisms. The MEMACK# signal is the common acknowledge signal for both mechanisms. Note that when MEMREQ# is asserted, FLSHREQ# is also asserted. Table 9 shows the relationship between MEMREQ# and FLSHREQ#.

Table 9. FLSHREQ#, MEMREQ#

FLSHREQ#	MEMREQ#	Meaning
1	1	Idle
0	1	Flush buffers pointing towards PCI to avoid ISA deadlock
1	0	Reserved
0	0	GAT mode, Guarantee PCI Bus immediate access to main memory

5.2.3.1 Flushing the System Posted Write Buffers

Once an ISA master or the DMA begins a cycle on the ISA Bus, the cycle can not be backed-off. It can only be held in wait states via IOCHRDY. In order to know the destination of ISA master cycles, the cycle needs to begin. However, after the cycle has started, no other device can intervene and gain ownership of the ISA Bus until the cycle has completed and arbitration is performed. A potential deadlock condition exists when an ISA originated cycle to the PCI Bus finds the PCI target inaccessible due to an interacting event that also requires the ISA Bus. To avoid this potential deadlock, all PCI posted write buffers in the system must be disabled and flushed before DACK can be returned. The buffers must remain disabled while the ISA Bus is occupied by an ISA master or the DMA.

When an ISA master or the DMA requests the ISA Bus, the SIO asserts FLSHREQ#. FLSHREQ# is an indication to the system to flush all posted write buffers pointing towards the PCI Bus. The SIO also flushes it's own Posted Write Buffer. Once the posted write buffers have been flushed and disabled, the system asserts MEMACK#. Once the SIO receives the MEMACK# acknowledgment signal, it asserts the DACK signal giving the requesting master the bus. FLSHREQ# stays active as long as the ISA master or DMA owns the ISA Bus.

5.2.3.2 Guaranteed Access Time Mode

Guaranteed Access Time (GAT) Mode is enabled/disabled via the PCI Arbiter Control Register. When this mode is enabled, the MEMREQ# and MEMACK# signals are used to guarantee that the ISA 2.5 μs IOCHRDY specification is not violated.

When an ISA master or DMA slave requests the ISA Bus (DREQ# active), the ISA Bus, the PCI Bus, and the memory bus must be arbitrated for and all three must be owned before the ISA master or DMA slave is granted the ISA Bus. After receiving the DREQ# signal from the ISA master or DMA slave, MEMREQ# and FLSHREQ# are asserted (FLSHREQ# is driven active, regardless of GAT mode being enabled or disabled). MEMREQ# is a request for direct access to main memory. MEMREQ# and FLSHREQ# will be asserted as long as the ISA master or the DMA owns the ISA Bus. When MEMACK# is received by the SIO (all posted write buffers are flushed and the memory bus is dedicated to the PCI interface), it will request the PCI Bus. When it is granted the PCI Bus, it asserts the DACK signal releasing the ISA Bus to the requesting master or the DMA.

The use of MEMREQ#, FLSHREQ#, and MEMACK# does not guarantee functionality with ISA masters that don't acknowledge IOCHRDY. These signals just guarantee the IOCHRDY inactive specification.

NOTE:
Usage of an external arbiter in GAT mode will require special logic in the arbiter.

5.2.4 RETRY THRASHING RESOLVE

When a PCI initiator's access is retried, the initiator releases the PCI Bus for a minimum of two PCI clocks and will then normally request the PCI Bus again. To avoid thrashing the bus with retry after retry, the PCI arbiter provides REQ# masking. The REQ# masking mechanism differentiates between SIO target retries and all other retries.

For initiators which were retried by the SIO as a target, the masked REQ# is flagged to be cleared upon RESUME# active. All other retries trigger the Master Retry Timer, if enabled. Upon expiration of this timer, the mask is cleared.

5.2.4.1 Resume Function (RESUME#)

The conditions under which the SIO forces a retry to a PCI master and will mask the REQ# are:

1. Any required buffer management
2. ISA Bus occupied by ISA master or DMA
3. The PCI to ISA Posted Write Buffer is full
4. The SIO is locked as a resource and LOCK# is asserted during the address process.

The RESUME# signal is pulsed whenever the SIO has retried a PCI cycle for one of the above reasons and that condition has passed. When RESUME# is asserted, the SIO will unmask the REQ#'s that are masked and flagged to be cleared by RESUME#.

If the internal arbiter is enabled, RESUME# is an internal signal. The RESUME# signal becomes visible as an output when the internal arbiter is disabled. This allows an external arbiter to optionally avoid retry thrashing associated with the SIO as a target. The RESUME# signal is asserted for one PCI clock.

5.2.4.2 Master Retry Timer

To re-enable a PCI master's REQ# which resulted in a retry to a slave other than the SIO, a SIO programmable Master Retry Timer has been provided. This timer can be programmed for 0 (disabled), 16, 32, or 64 PCICLKs. Once the SIO has detected that a PCI slave has forced a retry, the timer will be triggered and the corresponding master's REQ# will be masked. All subsequent PCI retries by this REQ# signal will by masked by the SIO. Expiration of this timer will unmask all of the masked requests. This timer has no effect on the request lines that have been masked due to a SIO retry.

If no other PCI masters are requesting the PCI Bus, all of the REQ#'s masked for the timer will be cleared and the timer will be reset. This is necessary to assist the host bridge in determining when to re-enable any disabled posted write buffers.

5.2.5 BUS PARKING

The SIO arbitration logic supplies a mechanism for PCI Bus parking. Parking is only allowed for the device which is tied to CPUREQ# (typically the system CPU). When bus parking is enabled, CPUGNT# will be asserted when no other agent is currently using or requesting the bus. This achieves the minimum PCI arbitration latency possible. Enabling of bus parking is achieved by programming the Arbiter Control Register. REQ0#, REQ1#, and the internal SIOREQ# are not allowed to park on the PCI Bus.

Upon assertion of CPUGNT# due to bus parking enabled and the PCI Bus idle, the CPU (or the parked agent) must ensure that AD[31:0], C/BE[3:0], and (one PCICLK later) PAR are driven. If bus parking is disabled, the SIO takes responsibility for driving the bus when it is idle.

5.2.6 BUS LOCK MODE

As an option, the SIO arbiter can be configured to run in Bus Lock Mode or Resource Lock Mode. The Bus Lock Mode is used to lock the entire PCI Bus. This may improve performance in some systems that frequently run quick read-modify-write cycles. Bus Lock Mode emulates the LOCK environment found in today's PC by restricting bus ownership when the PCI Bus is locked. With Bus Lock enabled, the arbiter recognizes a LOCK# being driven by any initiator and does not allow any other PCI initiator to be granted the PCI Bus until LOCK# and FRAME# are both negated indicating the master released lock. When Bus Lock is disabled, the default resource lock mechanism is implemented (normal resource lock) and a higher priority PCI initiator could intervene between the read and write cycles and run non-exclusive accesses to any unlocked resource.

5.2.7 POWER-UP CONFIGURATION

The SIO's arbiter is enabled if CPUREQ# is sampled high on the trailing edge of PCIRST#. When enabled, the arbiter is set in fixed priority mode 4 with CPU bus parking turned off. Fixed mode 4 guarantees that the CPU will be able to run accesses to the BIOS in order to configure the system, regardless of the state of the other REQ#'s. Note that the Host Bridge should drive CPUREQ# high during the trailing edge of PCIRST#. When the arbiter is enabled, the SIO acts as the central resource responsible for driving the AD[31:0], C/BE[3:0]#, and PAR signals when no one is granted the PCI Bus and the bus is idle. The SIO is always responsible for driving AD[31:0], C/BE[3:0]#, and PAR when it is granted the bus and as appropriate when it is the master of a transaction. After reset, if the arbiter is enabled, CPUGNT#, GNT0#, GNT1#, and the internal SIOGNT# will be driven based on the arbitration scheme and the asserted REQ#'s.

If an external arbiter is present in the system, the CPUREQ# signal should be tied low. When CPUREQ# is sampled low on the trailing edge of PCIRST#, the internal arbiter is disabled. When the internal arbiter is disabled, the SIO does not drive AD[31:0], C/BE[3:0]#, and PAR as the central resource. In this case, the SIO is only responsible for driving AD[31:0], C/BE[3:0]#, and PAR when it is granted the bus. If the SIO's arbiter is disabled, GNT0# becomes SIOREQ#, GNT1# becomes RESUME#, and REQ0# becomes SIOGNT#. This exposes the normally embedded SIO arbitration signals.

NOTE:
Usage of an external arbiter in GAT mode will require special logic in the arbiter.

5.3 ISA Interface

5.3.1 ISA INTERFACE OVERVIEW

The SIO incorporates a fully ISA Bus compatible master and slave interface. The SIO directly drives six ISA slots without external data or address buffers. The ISA interface also provides byte swap logic, I/O recovery support, wait-state generation, and SYSCLK generation.

The ISA interface supports the following types of cycles:

- PCI-initiated I/O and memory cycles to the ISA Bus.
- DMA compatible cycles between PCI memory and ISA I/O and between ISA I/O and ISA memory, DMA type "A", type "B", and type "F" cycles between PCI memory and ISA I/O.
- ISA Refresh cycles initiated by either the SIO or an external ISA master.
- ISA master-initiated memory cycles to the PCI Bus and ISA master-initiated I/O cycles to the internal SIO registers.

The refresh and DMA cycles are shown and described in Section 5.4.

5.3.2 SIO AS AN ISA MASTER

The SIO executes ISA cycles as an ISA master whenever a PCI initiated cycle is forwarded to the ISA Bus. The SIO also acts as an ISA master on behalf of DMA and refresh.

ISYSCLK is an internal 8 MHz clock.

5.3.3 SIO AS AN ISA SLAVE

The SIO operates as an ISA slave when:

- An ISA master accesses SIO internal registers.
- An ISA master accesses PCI memory on the PCI Bus.

5.3.3.1 ISA Master Accesses To SIO Registers

An ISA Bus master has access to SIO internal registers as shown in Table 19. An ISA master to SIO register cycle will always run as an 8-bit extended cycle (IOCHRDY will be held inactive until the cycle is completed).

Table 10. Arbitration Latency

Bus Condition	Arbitration Latency
Parked	0 PCICLKs for Agent 0, 2 PCICLKs for All Other
Not Parked	1 PCICLK for All Agents

5.3.3.2 ISA Master Accesses to PCI Resource

An ISA master can access PCI memory, but not I/O devices residing on the PCI Bus. The ISA/DMA address decoder determines which memory cycles should be directed towards the PCI Bus. During ISA master read cycles to the PCI Bus, the SIO will return all 1's if the PCI cycle is target-aborted or does not respond.

If the SIO is programmed for GAT mode, the SIO arbiter will not grant the ISA Bus before gaining ownership of both the PCI Bus and system memory. However, if the SIO is not programmed in this mode, the SIO does not need to arbitrate for the PCI Bus before granting the ISA Bus to the ISA master. For more details on the arbitration, refer to Section 5.2.2.

All cycles forwarded to a PCI resource will run as 16-bit extended cycles (i.e. IOCHRDY will be held inactive until the cycle is completed).

Because the ISA bus size is different from the PCI bus size, the data steering logic inside the SIO is responsible for steering the data to the correct byte lanes on both buses, and assembling/disassembling the data as necessary.

5.3.4 ISA MASTER TO ISA SLAVE SUPPORT

During ISA master cycles to ISA slaves, the SIO drives several signals to support the transfer:

BALE:

This signal is driven high while the ISA master owns the ISA Bus.

AEN:

This signal is driven low while the ISA master owns the ISA Bus.

SMEMR# and **SMEMW#**:

These signals are driven active by the SIO whenever the ISA master drives a memory cycle to an address below 1 Mb.

Utility Bus Buffer Control Signals and Chip Select Signals:

These signals are driven active as appropriate whenever an ISA master accesses devices on the Utility Bus. For more details, see Section 5.9.

Data Swap Logic:

The data swap logic inside the SIO is activated as appropriate to swap data between the even and odd byte lanes. This is discussed in further detail in Section 5.3.5.

5.3.5 DATA BYTE SWAPPING

The data swap logic is integrated in the SIO. For slaves that reside on the ISA Bus, data swapping is performed if the slave (I/O or memory) and ISA bus master (or DMA) sizes differ and the upper (odd) byte of data is being accessed. Table 11 shows when data swapping is provided during DMA. Table 12 shows when data swapping is provided during ISA master cycles to 8-bit ISA slaves.

Table 11. DMA Data Swap

DMA I/O Device Size	ISA Memory Slave Size	Swap	Comments I/O	⟷	Memory
8-Bit	8-Bit	No	SD[7:0]	⟷	SD[7:0]
8-Bit	16-Bit	Yes	SD[7:0]	⟷	SD[7:0]
			SD[7:0]	⟷	SD[15:8]
16-Bit	8-Bit	No	Not Supported		
16-Bit	16-Bit	No	SD[15:0]	⟷	SD[15:0]

The SIO monitors the SBHE# and SA0 signals to determine when to swap the data. The SIO ensures that the data is placed on the appropriate byte lane.

Table 12. 16-Bit Master to 8-Bit Slave Data Swap

SBHE#	SA0	SD[15:8]	SD[7:0]	Comments
0	0	Odd	Even	Word Transfer (data swapping not required)
0	1	Odd	Even	Byte Swap(1,2)
1	0	—	Even	Byte Transfer (data swapping not required)
1	1	—	—	Not Allowed

NOTES:
1. For ISA master read cycles, the SIO swaps the data from the lower byte to the upper byte.
2. For ISA master write cycles, the SIO swaps the data from the upper byte to the lower byte.

5.3.6 ISA CLOCK GENERATION

The SIO generates the ISA system clock (SYSCLK). SYSCLK is a divided down version of the PCICLK (see Table 13). The clock divisor value is programmed through the ISA Clock Divisor Register.

Table 13. SYSCLK Generation from PCICLK

PCICLK (MHz)	Divisor (Programmable)	SYSCLK (MHz)
25	3	8.33
33	4 (default)	8.33

NOTE:
For PCI frequencies less than 33 MHz (not including 25 MHz), a clock divisor value must be selected that ensures that the ISA Bus frequency does not violate the 6 MHz to 8.33 MHz SYSCLK specification.

5.3.7 WAIT STATE GENERATION

The SIO will add wait states to the following cycles, if IOCHRDY is sampled active low. Wait states will be added as long as IOCHRDY is low.

- During Refresh and SIO master cycles (not including DMA) to the ISA Bus.
- During DMA compatible transfers between ISA I/O and ISA memory only.

For ISA master cycles targeted for the SIO's internal registers or PCI memory, the SIO will always extend the cycle by driving IOCHRDY low until the transaction is complete.

The SIO will shorten the following cycles, if ZEROWS# is sampled active.

- During SIO master cycles (not including DMA) to 8-bit and 16-bit ISA memory.
- During SIO master cycles (not including DMA) to 8-bit ISA I/O only.

For ISA master cycles targeted for the SIO's internal registers or PCI memory, the SIO will not assert ZEROWS#.

NOTE:
If IOCHRDY and ZEROWS# are sampled active at the same time, IOCHRDY will take precedence and wait-states will be added.

5.3.8 I/O RECOVERY

The I/O recovery mechanism in the SIO is used to add additional recovery delay between PCI originated 8-bit and 16-bit I/O cycles to the ISA Bus. The SIO automatically forces a minimum delay of four SYSCLKs between back-to-back 8- and 16-bit I/O cycles to the ISA Bus. This delay is measured from the rising edge of the I/O command (IOR# or IOW#) to the falling edge of the next BALE. If a delay of greater than four SYSCLKs is required, the ISA I/O Recovery Time Register can be programmed to increase the delay in increments of SYSCLKs. No additional delay is inserted for back-to-back I/O "sub cycles" generated as a result of byte assembly or disassembly .

5.4 DMA Controller

5.4.1 DMA CONTROLLER OVERVIEW

The DMA circuitry incorporates the functionality of two 82C37 DMA controllers with seven independently programmable channels (Channels 0–3 and Channels 5–7). DMA Channel 4 is used to cascade the two controllers and will default to cascade mode in the DMA Channel Mode (DCM) Register. In addition to accepting requests from DMA slaves, the DMA controller also responds to requests that are initiated by software. Software may initiate a DMA service request by setting any bit in the DMA Channel Request Register to a 1. The DMA controller for Channels 0–3 is referred to as "DMA-1" and the controller for Channels 4–7 is referred to as "DMA-2".

Each DMA channel may be programmed for 8- or 16-bit DMA device size and ISA-compatible, Type "A", Type "B", or Type "F" transfer timing. Each DMA channel defaults to the compatible settings for DMA device size: channels [3:0] default to 8-bit, count-by-bytes transfers, and channels [7:5] default to 16-bit, count-by-words (address shifted) transfers. The SIO provides the timing control and data size translation necessary for the DMA transfer between the PCI and the ISA Bus. ISA-compatible is the default transfer timing.

Full 32-bit addressing is supported as an extension of the ISA-compatible specification. Each channel includes a 16-bit ISA compatible Current Register which holds the 16 least-significant bits of the 32-bit address, and an ISA compatible Low Page Register which contains the eight second most significant bits. An additional High Page Register contains the eight most significant bits of the 32-bit address.

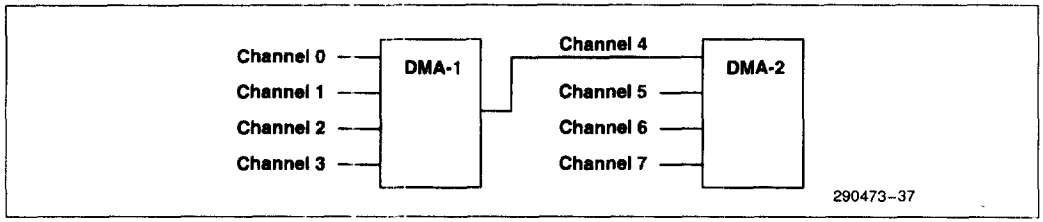


Figure 7. Internal DMA Controller

The DMA controller also features refresh address generation, and auto-initialization following a DMA termination.

The DMA controller receives commands from the ISA Bus arbiter to perform either DMA cycles or refresh cycles. The arbiter determines which requester from among the requesting DMA slaves, the PCI Bus, and refresh should have the bus.

The DMA controller is at any time either in master mode or slave mode. In master mode, the DMA controller is either servicing a DMA slave's request for DMA cycles, or allowing a 16-bit ISA master to use the bus via a cascaded DREQ signal. In slave mode, the SIO monitors both the ISA Bus and the PCI, decoding and responding to I/O read and write commands that address its registers.

Note that a DMA device (I/O device) is always on the ISA Bus, but the memory device is either on the ISA or PCI Bus. If the memory is decoded to be on the ISA Bus, then the DMA cycle will run as a compatible cycle. If the memory is decoded to be on the PCI Bus, the cycle can run as compatible, "A", "B", or "F" type. The ISA controller will not drive a valid address for type "A", "B", and "F" DMA transfers on the ISA Bus.

When the SIO is running a DMA cycle in compatible timing mode, the SIO will drive the MEMR# or MEMW# strobes if the address is less than 16 MBytes (00000000h–00FFFFFFh). These memory strobes will be generated regardless of whether the cycle is decoded for PCI or ISA memory. The SMEMR# and SMEMW# will be generated if the address is less than 1 MBytes (00000000h–000FFFFFh). If the address is greater than 16 MBytes (01000000h–FFFFFFFFh) the MEMR# or MEMW# strobe will not be generated in order to avoid aliasing problems. For type "A", "B", and "F" timing mode DMA cycles, the SIO will only generate the MEMR# or MEMW# strobe when the address is decoded for ISA memory. When this occurs, the cycle converts to compatible mode timing.

During DMA cycles, the ISA controller drives AEN high to prevent the I/O devices from misinterpreting the DMA cycle as a valid I/O cycle. The BALE signal is also driven high during DMA cycles. Also, during DMA memory read cycles to the PCI Bus, the SIO will return all 1's to the ISA Bus if the PCI cycle is either target-aborted or does not respond.

Further details can be found in the 82C37 data sheet.

5.4.2 DMA TIMINGS

ISA Compatible timing is provided for DMA slave devices. Three additional timings are provided for I/O slaves capable of running at faster speeds. These timings are referred to as Type "A", Type "B", and Type "F".

5.4.2.1 Compatible Timing

Compatible timing runs at 8 SYSCLKs during the repeated portion of a Block or Demand mode transfer.

5.4.2.2 Type "A" Timing

Type "A" timing is provided to allow shorter cycles to PCI memory. Type "A" timing runs at 6 SYSCLKs (720 ns/cycle) during the repeated portion of a block or demand mode transfer. This timing assumes an 8.33 MHz SYSCLK. Type "A" timing varies from compatible timing primarily in shortening the memory operation to the minimum allowed by system memory. The I/O portion of the cycle (data setup on write, I/O read access time) is the same as with compatible cycles. The actual active command time is shorter, but it is expected that the DMA devices which provide the data access time or write data setup time should not require excess IOR# or IOW# command active time. Because of this, most ISA DMA devices should be able to use type "A" timing.

5.4.2.3 Type "B" Timing

Type "B" timing is provided for 8-/16-bit ISA DMA devices which can accept faster I/O timing. Type "B" only works with PCI memory. Type "B" timing runs at 5 SYSCLKs (600 ns/cycle) during the repeated portion of a Block or Demand mode transfer. This timing assumes an 8.33 MHz SYSCLK. Type "B" timing requires faster DMA slave devices than compatible timing in that the cycles are shortened so that the data setup time on I/O write cycles is shortened and the I/O read access time is required to be faster. Some of the current ISA devices should be able to support type "B" timing, but these will probably be more recent designs using relatively fast technology.

5.4.2.4 Type "F" Timing

Type "F" timing provides high performance DMA transfer capability. These transfers are mainly for fast I/O devices (i.e. IDE devices). Type "F" timing runs at 3 SYSCLKs (360 ns/cycle) during the repeated portion of a Block or Demand mode transfer.

5.4.2.5 DREQ and DACK# Latency Control

The SIO DMA arbiter maintains a minimum DREQ to DACK# latency on DMA channels programmed to operate in compatible timing mode. This is to support older devices such as the 8272A. The DREQs are delayed by eight SYSCLKs prior to being seen by the arbiter logic. Software requests will not have this minimum request to DACK# latency.

5.4.3 ISA BUS/DMA ARBITRATION

The ISA Bus arbiter evaluates requests for the ISA Bus coming from several different sources. The DMA unit, the refresh counter, and the PCI Bus (primarily the Host CPU) may all request access to the ISA Bus. Additionally, 16-bit ISA masters may request the bus through a cascaded DMA channel.

The SIO ISA arbiter uses a three-way rotating priority arbitration method. At each level, the devices which are considered equal are given a rotating priority. On a fully loaded bus, the order in which the devices are granted bus access is independent of the order in which they assert a bus request. This is because devices are serviced based on their position in the rotation. The arbitration scheme assures that DMA channels access the bus with minimal latency.

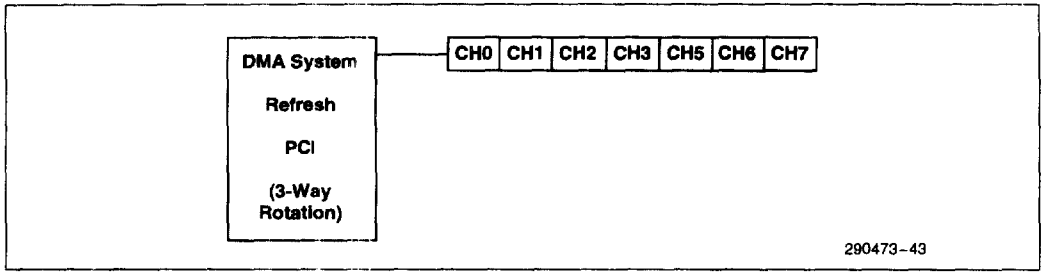


Figure 8. ISA Arbiter with DMA in Fixed Priority

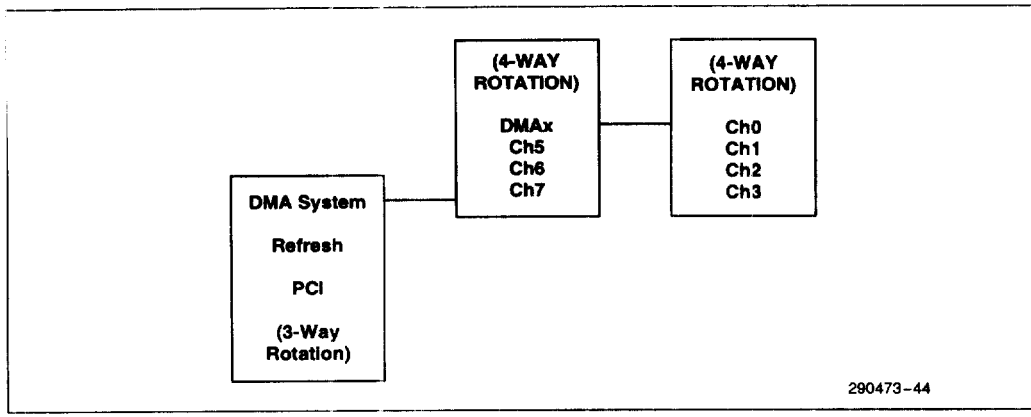


Figure 9. ISA Arbiter with DMA in Rotating Priority

1

5.4.3.1 Channel Priority

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

For prioritization purposes, the source of the DMA request is transparent. DMA I/O slaves normally assert their DREQ line to arbitrate for DMA service. However, a software request for DMA service can be presented through each channel's DMA Request Register. A software request is subject to the same prioritization as any hardware request. Please see the detailed register description in Section 4.2.4 for Request Register programming information.

Fixed Priority

The initial fixed priority structure is as follows:

Table 14. Initial Fixed Priority Structure

High Priority . . . . . Low Priority
(0, 1, 2, 3), 5, 6, 7

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

Rotating Priority

Rotation allows for "fairness" in priority resolution. The priority chain rotates so that the last channel serviced is assigned the lowest priority in the channel group (0–3, 5–7).

Channels 0–3 rotate as a group of 4. They are always placed between Channel 5 and Channel 7 in the priority list.

Channel 5–7 rotate as part of a group of 4. That is, channels (5–7) form the first three positions in the rotation, while channel group (0–3) comprises the fourth position in the arbitration.

Table 15 demonstrates rotation priority.

Table 15. Rotating Priority Example

Programmed Mode	Action	Priority High .....Low
Group (0–3) is in Rotation Mode Group (4–7) is in Fixed Mode	1) Initial Setting	(0, 1, 2, 3), 5, 6, 7
	2) After Servicing Channel 2	(3, 0, 1, 2), 5, 6, 7
	3) After Servicing Channel 3	(0, 1, 2, 3), 5, 6, 7
Group (0–3) is in Rotation Mode Group (4–7) is in Rotation Mode	1) Initial Setting	(0, 1, 2, 3), 5, 6, 7
	2) After Servicing Channel 0	5, 6, 7, (1, 2, 3, 0)
	3) After Servicing Channel 5	6, 7, (1, 2, 3, 0), 5
	4) After servicing Channel 6	7, (1, 2, 3, 0), 5, 6
	5) After servicing Channel 7	(1, 2, 3, 0), 5, 6, 7

NOTE:
The first servicing of channel 0 caused double rotation.

5.4.3.2 DMA Preemption In Performance Timing Modes

A DMA slave device that is not programmed for compatible timing will be preempted from the bus by another device that requests use of the bus. This will occur, regardless of the priority of the pending request. For DMA devices not using compatible timing mode, the DMA controller stops the DMA transfer and releases the bus within 32 BCLK (4 μs) of a preemption. Upon the expiration of the 4 μs timer, the DACK will be inactivated after the current DMA cycle has completed. The bus will then be arbitrated for and granted to the highest priority requester. This feature allows flexibility in programming the DMA for long transfer sequences in a performance timing mode while guaranteeing that vital system services such as refresh are allowed access to the ISA Bus.

The 4 μs timer is not used in compatible timing mode. It is only used for DMA channels programmed for Type "A", Type "B", or Type "F" timing. It is also not used for 16-bit ISA masters cascaded through the DMA DREQ lines.

If the DMA channel that was preempted by the 4 μs timer was operating in Block Mode, an internal bit will be set so that the channel will be arbitrated for again, independent of the state of DREQ.

5.4.3.3. Arbitration during Non-Maskable Interrupts

If a non-maskable interrupt (NMI) is pending, and the CPU is requesting the bus, then the DMA controller will be bypassed each time it comes up for rotation. This will give the CPU the bus bandwidth it requires to process the interrupt as fast as possible.

5.4.3.4 Programmable Guaranteed Access Time Mode (GAT Mode)

The PCI Arbiter Register contains a bit for configuring the SIO in "Guaranteed Access Time Mode" (GAT Mode). This mode guarantees that the 2.5 μs CHRDY time-out specification for ISA masters running cycles to PCI will not be exceeded. When an ISA master or DMA slave arbitrates for the ISA Bus, and the SIO is configured in Guaranteed Access Time Mode, the MEMREQ# pin will be asserted by the PCI arbiter in order to gain ownership of main memory. The arbitration for the PCI and then the main memory bus must be completed prior to granting the DACK# to the ISA master or DMA slave. A MEMACK# signal to the SIO indicates that the SIO now owns main memory and can grant the DACK# to the ISA master or DMA slave. A detailed description is contained in Section 5.2.3.2.

5.4.4 REGISTER FUNCTIONALITY

1

Please see Section 4.2 for detailed information on register programming, bit definitions, and default values/functions of the DMA registers after a PCIRST#.

DMA Channel 4 is used to cascade the two DMA controllers together and should not be programmed for any mode other than cascade. The DMA Channel Mode Register for channel 4 will default to cascade mode. Special attention should also be take when programming the Command and Mask Registers as related to channel 4.

5.4.4.1 Address Compatibility Mode

Whenever the DMA is operating in address compatibility mode, the addresses do not increment or decrement through the High and Low Page Registers, and the High Page Register is set to 00h. This is compatible with the 82C37 and Low Page Register implementation used in the PC/AT. This mode is set when any of the lower three address bytes of a channel are programmed. If the upper byte of a channel's address is programmed last, the channel will go into extended address mode. In this mode, the high byte may be any value and the address will increment or decrement through the entire 32-bit address.

After PCIRST# is negated, all channels will be set to address compatibility mode. The DMA Master Clear command will also reset the proper channels to address compatibility mode.

5.4.4.2 Summary of DMA Transfer Sizes

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

5.4.4.3 Address Shifting when Programmed for 16-Bit I/O Count by Words

To maintain compatibility with the implementation of the DMA in the PC/AT which used the 82C37, the DMA will shift the addresses when the DMA Channel Extended Mode Register is programmed for, or defaulted to, transfers to/from a 16-bit device count-by-words. Note that the least significant bit of the Low Page Register is dropped in 16-bit shifted mode. When programming the Current Address Register when the DMA channel is in this mode, the Current Address must be programmed to an even address with the address value shifted right by one bit. The address shifting is shown in Table 17.

5.4.4.4 Autoinitialize

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

Table 16. DMA Transfer Size

DMA Device Date Size and Word Count	Current Byte/Word Count Register	Current Address Increment/Decrement
8-Bit I/O, Count by Bytes	Bytes	1
16-Bit I/O, Count by Words (Address Shifted)	Words	1
16-Bit I/O, Count by Bytes	Bytes	2

Table 17. Address Shifting in 16-Bit I/O DMA Transfers

Output Address	8-Bit I/O Programmed Address	16-Bit I/O Programmed Address (Shifted)	16-Bit I/O Programmed Address (No Shift)
A0	A0	0	A0
A[16:1]	A[16:1]	A[15:0]	A[16:1]
A[31:17]	A[31:17]	A[31:17]	A[31:17]

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

5.4.5 SOFTWARE COMMANDS

There are three additional special software commands which can be executed by the DMA controller. The three software commands are:

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

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

5.4.5.1 Clear Byte Pointer Flip-Flop

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

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

5.4.5.2 DMA Master Clear

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

There are two independent master clear commands, 0Dh which acts on channels 0–3, and 0DAh which acts on channels 4–7.

5.4.5.3 Clear Mask Register

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

5.4.6 TERMINAL COUNT/EOP SUMMARY

This is a summary of the events that will happen as a result of a terminal count or external EOP when running DMA in various modes. (See Table 18.)

5.4.7 ISA REFRESH CYCLES

Refresh cycles are generated by two sources: the refresh controller inside the SIO component or by ISA bus masters other than the SIO. The ISA bus controller will enable the address lines SA[15:0] so that when MEMR# goes active, the entire ISA system memory is refreshed at one time. Memory slaves on the ISA Bus must not drive any data onto the data bus during the refresh cycle.

Counter 1 in the timer register set should be programmed to provide a request for refresh about every 15 μs.

Table 18. Terminal Count/EOP Summary Table

Conditions AUTOINIT	No		Yes	
Event				
Word Counter Expired	Yes	X	Yes	X
EOP Input	X	Asserted	X	Asserted
Result				
Status TC	set	set	set	set
Mask	set	set	—	—
SW Request	clr	clr	clr	clr
Current Register	—	—	load	load

NOTES:
load = load current from base
— = no change
X = don't care
clr = clear

5.4.8 SCATTER/GATHER DESCRIPTION

Scatter/Gather (S/G) provides the capability of transferring multiple buffers between memory and I/O without CPU intervention. In Scatter/Gather, the DMA can read the memory address and word count from an array of buffer descriptors, located in system memory (ISA or PCI), called the Scatter/Gather Descriptor (SGD) Table. This allows the DMA controller to sustain DMA transfers until all of the buffers in the SGD Table are transferred.

The S/G Command and Status Registers are used to control the operational aspect of S/G transfers. The SGD Table Pointer Register holds the address of the next buffer descriptor in the SGD Table.

The next buffer descriptor is fetched from the SGD Table by a DMA read transfer. DACK# will not be asserted for this transfer because the IO device is the SIO itself. The SIO will fetch the next buffer descriptor from either PCI memory or ISA memory, depending on where the SGD Table is located. If the SGD table is located in PCI memory, the memory read will use the line buffer to temporarily store the PCI read before loading it into the DMA S/G registers. The line buffer mode (8 byte or single transaction) for the S/G fetch operation will be the same as what is set for all DMA operations. If set in 8 byte mode, the SGD Table fetches will be PCI burst memory reads. The SGD Table PCI cycle fetches are subject to all types of PCI cycle termination (retry, disconnect, target-abort, master-abort). The fetched SGD Table data is subject to normal line buffer coherency management and invalidation. EOP will be asserted at the end of the complete link transfer.

To initiate a typical DMA Scatter/Gather transfer between memory and an I/O device, the following steps are required:

1. Software prepares a SGD Table in system memory. Each SGD is 8 bytes long and consists of an address pointer to the starting address and the transfer count of the memory buffer to be transferred. In any given SGD Table, two consecutive SGDs are offset by 8 bytes and are aligned on a 4-byte boundary.

   Each Scatter/Gather Descriptor for the linked list contains the following information:

   a. Memory Address (buffer start) 4 bytes

   b. Transfer Size (buffer size) 2 bytes

   c. End of Link List 1 bit (MSB)

2. Initialize the DMA Channel Mode and DMA Channel Extended Mode Registers with transfer specific information like 8-/16-bit I/O device, Transfer Mode, Transfer Type, etc.
3. Software provides the starting address of the SGD Table by loading the SGD Table Pointer Register.
4. Engage the Scatter/Gather function by writing a Start command to the S/G Command Register.
5. The Mask register should be cleared as the last step of programming the DMA register set. This is to prevent the DMA from starting a transfer with a partially loaded command description.
6. Once the register set is loaded and the channel is unmasked, the DMA will generate an internal request to fetch the first buffer from the SGD Table.

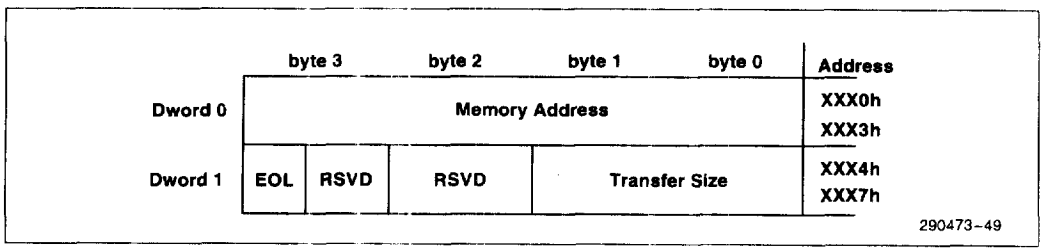


Figure 10. SGD Format

After the above steps are finished, the DMA will then respond to DREQ or software requests. The first transfer from the first buffer moves the memory address and word count from the Base register set to the Current register set. As long as S/G is active and the Base register set is not loaded and the last buffer has not been fetched, the channel will generate a request to fetch a reserve buffer into the Base register set. The reserve buffer is loaded to minimize latency problems going from one buffer to another. Fetching a reserve buffer has a lower priority than completing DMA transfers for the channel.

The DMA controller will terminate a Scatter/Gather cycle by detecting an End of List (EOL) bit in the SGD Table. After the EOL bit is detected, the channel transfers the buffers in the Base and Current register sets, if they are loaded. At terminal count the channel asserts EOP or IRQ13, depending on its programming and set the terminate bit in the S/G Status Register. If the channel asserted IRQ13, then the appropriate bit is set in the S/G Interrupt Status Register. The active bit in the S/G Status Register will be reset and the channel's Mask bit will be set.

5.5 Address Decoding

The SIO contains two address decoders; one to decode PCI master cycles and one to decode DMA/ISA master cycles. Two decoders are required to support the PCI and ISA Buses running concurrently. The PCI address decoder decodes the address from the multiplexed PCI address/data bus. The DMA/ISA master address decoder decodes the address from the ISA address bus for DMA and ISA master cycles. The address decoders determine how the cycle is handled.

1

5.5.1 PCI ADDRESS DECODER

PCI address decoding is always a function of AD[31:2]. The information contained in the two low order bits (AD[1:0]) varies for memory, I/O, and configuration cycles.

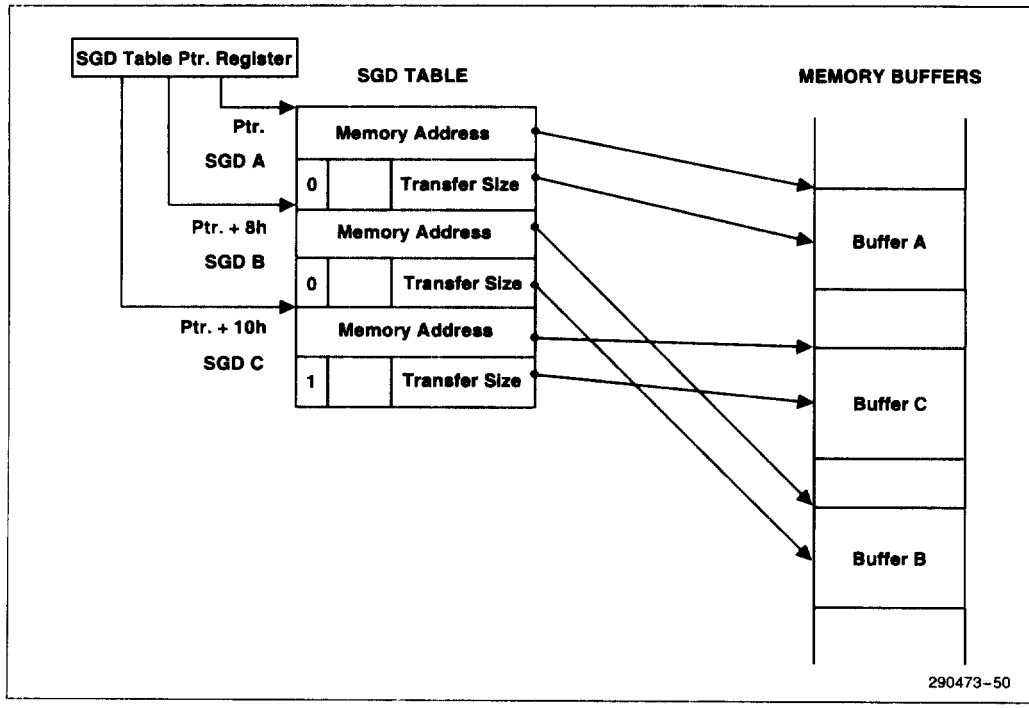


Figure 11. Link List Example

For I/O cycles, AD[31:0] are all decoded to provide a byte address. The byte enables determine which byte lanes contain valid data. The SIO requires that PCI accesses to byte-wide internal registers must assert only one byte enable.

For memory cycles, accesses are decoded as Dword accesses. This means that AD[1:0] are ignored for decoding memory cycles. The byte enables are used only to determine which byte lanes contain valid data.

For configuration cycles, DEVSEL# is a function of IDSEL# and AD[1:0]. DEVSEL# is generated only when AD[1:0] are both zero. If either AD[1:0] are non-zero, the cycle is ignored by the SIO. Individual bytes of a configuration register can be accessed with the byte enables. A particular configuration register is selected using AD[7:2]. Again, the byte enables determine which byte lanes contain valid data.

All PCI cycles decoded in one of the following ways result in the SIO generating DEVSEL#. The PCI master cycle decoder decodes the following addresses based on the settings of the relevant configuration registers:

SIO I/O Addresses: Positively decodes I/O addresses for registers contained within the SIO (exceptions: 60h, 92h, 3F2h, 372h, and F0h).

BIOS Memory Space: Positively decodes BIOS memory space.

MEMCS# Address Decoding: Decodes memory addresses that reside on the other side of the Host bridge and generates the MEMCS# signal. (SIO does not generate DEVSEL# in this case). The address range(s) used for this decoding is selected via the MEMCSCON, MEMCSBOH, MEMCSTOH, MEMCSTOM, MAR1, MAR2, and MAR3 Registers (see Section 4.1).

Subtractively Decoding Cycles to ISA: Subtractively decodes cycles to the ISA Bus. Accesses to registers 60h, 92h, 3F2h, 372h, and F0h are also subtractively decoded to the ISA Bus.

One of the PCI requirements is that, upon power-up, PCI agents do not respond to any address. Typically, the only access to a PCI agent is through the IDSEL configuration mechanism until it is enabled through configuration. The SIO is an exception to this, since it controls access to the BIOS boot code. All addresses decoded by the PCI address decoder, that are enabled after chip reset, are accessible immediately after power-up.

5.5.1.1 SIO I/O Addresses

These addresses are the internal, non-configuration SIO register locations and are shown in the SIO Address Decoding Table, Table 19. These addresses are fixed. Note that the Configuration Registers, listed in Table 3, are accessed with PCI configuration cycles as described in Section 5.1.2.5

In general, PCI accesses to the internal SIO registers will not be broadcast to the ISA Bus. However, PCI accesses to addresses 70h, 60h, 92h, 3F2h, 372h, and F0h are exceptions. Read and write accesses to these SIO locations are broadcast onto the ISA Bus. PCI master accesses to SIO registers will be retried if the ISA Bus is owned by an ISA master or the DMA controller. All of the registers are 8 bit registers. Accesses to these registers must be 8 bit accesses. Target-abort is issued by the SIO when the internal SIO non-configuration registers are the target of a PCI master I/O cycle and more than one byte enable is active. Refer to Table 19 for the SIO Address Decoding Map.

Accesses to the BIOS Timer Register (78h–7Bh) are broadcast to the ISA bus only if this register is disabled. If this register is enabled, the cycle is not broadcast to the ISA bus.

The address decoding logic includes the read/write cycle type. For example, read cycles to write only registers are not positively decoded and get forwarded to the ISA bus via subtractive decoding.

Table 19. SIO Address Decoding

Address	Address				Type	Name	Block
	FEDC	BA98	7654	3210			
0000h	0000	0000	000x	0000	r/w	DMA1 CH0 Base and Current Address	DMA
0001h	0000	0000	000x	0001	r/w	DMA1 CH0 Base and Current Count	DMA
0002h	0000	0000	000x	0010	r/w	DMA1 CH1 Base and Current Address	DMA
0003h	0000	0000	000x	0011	r/w	DMA1 CH1 Base and Current Count	DMA
0004h	0000	0000	000x	0100	r/w	DMA1 CH2 Base and Current Address	DMA
0005h	0000	0000	000x	0101	r/w	DMA1 CH2 Base and Current Count	DMA
0006h	0000	0000	000x	0110	r/w	DMA1 CH3 Base and Current Address	DMA
0007h	0000	0000	000x	0111	r/w	DMA1 CH3 Base and Current Count	DMA
0008h	0000	0000	000x	1000	r/w	DMA1 Status(r) Command(w) Register	DMA
0009h	0000	0000	000x	1001	wo	DMA1 Write Request Register	DMA
000Ah	0000	0000	000x	1010	wo	DMA1 Write Single Mask Bit	DMA
000Bh	0000	0000	000x	1011	wo	DMA1 Write Mode Register	DMA
000Ch	0000	0000	000x	1100	wo	DMA1 Clear Byte Pointer	DMA
000Dh	0000	0000	000x	1101	wo	DMA1 Master Clear	DMA
000Eh	0000	0000	000x	1110	wo	DMA1 Clear Mask Register	DMA
000Fh	0000	0000	000x	1111	r/w	DMA1 Read/Write All Mask Register Bits	DMA
0020h	0000	0000	001x	xx00	r/w	INT 1 Control Register	PIC
0021h	0000	0000	001x	xx01	r/w	INT 1 Mask Register	PIC
0040h	0000	0000	010x	0000	r/w	Timer Counter 1—Counter 0 Count	TC
0041h	0000	0000	010x	0001	r/w	Timer Counter 1—Counter 1 Count	TC
0042h	0000	0000	010x	0010	r/w	Timer Counter 1—Counter 2 Count	TC
0043h	0000	0000	010x	0011	wo	Timer Counter 1 Command Mode Register	TC
0060h	0000	0000	0110	0000	ro	Reset UBus IRQ12	Control
0061h	0000	0000	0110	0xx1	r/w	NMI Status and Control	Control
0070h	0000	0000	0111	0xx0	wo	CMOS RAM Address and NMI Mask Register	Control
0078h(1)	0000	0000	0111	10xx	r/w	BIOS Timer	TC

Table 19. SIO Address Decoding (Continued)

Address	Address				Type	Name	Block
	FEDC	BA98	7654	3210			
0080h	0000	0000	100x	0000	r/w	DMA Page Register (Reserved)	DMA
0081h	0000	0000	100x	0001	r/w	DMA Channel 2 Page Register	DMA
0082h	0000	0000	1000	0010	r/w	DMA Channel 3 Page Register	DMA
0083h	0000	0000	100x	0011	r/w	DMA Channel 1 Page Register	DMA
0084h	0000	0000	100x	0100	r/w	DMA Page Register (Reserved)	DMA
0085h	0000	0000	100x	0101	r/w	DMA Page Register (Reserved)	DMA
0086h	0000	0000	100x	0110	r/w	DMA Page Register (Reserved)	DMA
0087h	0000	0000	100x	0111	r/w	DMA Channel 0 Page Register	DMA
0088h	0000	0000	100x	0100	r/w	DMA Page Register (Reserved)	DMA
0089h	0000	0000	100x	1001	r/w	DMA Channel 6 Page Register	DMA
008Ah	0000	0000	100x	1010	r/w	DMA Channel 7 Page Register	DMA
008Bh	0000	0000	100x	1011	r/w	DMA Channel 5 Page Register	DMA
008Ch	0000	0000	100x	1100	r/w	DMA Page Register (Reserved)	DMA
008Dh	0000	0000	100x	1101	r/w	DMA Page Register (Reserved)	DMA
008Eh	0000	0000	100x	1110	r/w	DMA Page Register (Reserved)	DMA
008Fh	0000	0000	100x	1111	r/w	DMA Low Page Register Refresh	DMA
0090h	0000	0000	100x	0000	r/w	DMA Page Register (Reserved)	DMA
0092h	0000	0000	1001	0010	r/w	System Control Port	Control
0094h	0000	0000	100x	0100	r/w	DMA Page Register (Reserved)	DMA
0095h	0000	0000	100x	0101	r/w	DMA Page Register (Reserved)	DMA
0096h	0000	0000	100x	0110	r/w	DMA Page Register (Reserved)	DMA
0098h	0000	0000	100x	1000	r/w	DMA Page Register (Reserved)	DMA
009Ch	0000	0000	100x	1100	r/w	DMA Page Register (Reserved)	DMA
009Dh	0000	0000	100x	1101	r/w	DMA Page Register (Reserved)	DMA
009Eh	0000	0000	100x	1110	r/w	DMA Page Register (Reserved)	DMA
009Fh	0000	0000	100x	1111	r/w	DMA low page Register Refresh	DMA
00A0h	0000	0000	101x	xx00	r/w	INT 2 Control Register	PIC
00A1h	0000	0000	101x	xx01	r/w	INT 2 Mask Register	PIC
00B2h	0000	0000	1011	0010	r/w	Advanced Power Management Control Port	PM
00B3h	0000	0000	1011	0011	r/w	Advanced Power Management Status Port	PM
00C0h	0000	0000	1100	000x	r/w	DMA2 CH0 Base and Current Address	DMA

Table 19. SIO Address Decoding (Continued)

Address	Address				Type	Name	Block
	FEDC	BA98	7654	3210			
00C2h	0000	0000	1100	001x	r/w	DMA2 CH0 Base and Current Count	DMA
00C4h	0000	0000	1100	010x	r/w	DMA2 CH1 Base and Current Address	DMA
00C6h	0000	0000	1100	011x	r/w	DMA2 CH1 Base and Current Count	DMA
00C8h	0000	0000	1100	100x	r/w	DMA2 CH2 Base and Current Address	DMA
00CAh	0000	0000	1100	101x	r/w	DMA2 CH2 Base and Current Count	DMA
00CCh	0000	0000	1100	110x	r/w	DMA2 CH3 Base and Current Address	DMA
00CEh	0000	0000	1100	111x	r/w	DMA2 CH3 Base and Current Count	DMA
00D0h	0000	0000	1101	000x	r/w	DMA2 Status(r) Command(w) Register	DMA
00D2h	0000	0000	1101	001x	wo	DMA2 Write Request Register	DMA
00D4h	0000	0000	1101	010x	wo	DMA2 Write Single Mask Bit	DMA
00D6h	0000	0000	1101	011x	wo	DMA2 Write Mode Register	DMA
00D8h	0000	0000	1101	100x	wo	DMA2 Clear Byte Pointer	DMA
00DAh	0000	0000	1101	101x	wo	DMA2 Master Clear	DMA
00DCh	0000	0000	1101	110x	wo	DMA2 Clear Mask Register	DMA
00DEh	0000	0000	1101	111x	r/w	DMA2 Read/Write All Mask Register Bits	DMA
00F0h	0000	0000	1111	0000	wo	Coprocessor Error	Control
0372h	0000	0011	0111	0010	wo	Secondary Floppy Disk Digital Output Reg.	Control
03F2h	0000	0011	1111	0001	wo	Primary Floppy Disk Digital Output Reg.	Control
040Ah	0000	0100	0000	1010	ro	Scatter/Gather Interrupt Status Register	DMA
040Bh	0000	0100	0000	1011	wo	DMA1 Extended Mode register	DMA
0410h(1)	0000	0100	0001	0000	wo	CH0 Scatter/Gather Command	DMA
0411h(1)	0000	0100	0001	0001	wo	CH1 Scatter/Gather Command	DMA
0412h(1)	0000	0100	0001	0010	wo	CH2 Scatter/Gather Command	DMA
0413h(1)	0000	0100	0001	0011	wo	CH3 Scatter/Gather Command	DMA
0415h(1)	0000	0100	0001	0101	wo	CH5 Scatter/Gather Command	DMA
0416h(1)	0000	0100	0001	0110	wo	CH6 Scatter/Gather Command	DMA
0417h(1)	0000	0100	0001	0111	wo	CH7 Scatter/Gather Command	DMA
0418h(1)	0000	0100	0001	1000	ro	CH0 Scatter/Gather Status	DMA
0419h(1)	0000	0100	0001	1001	ro	CH1 Scatter/Gather Status	DMA
041Ah(1)	0000	0100	0001	1010	ro	CH2 Scatter/Gather Status	DMA
041Bh(1)	0000	0100	0001	1011	ro	CH3 Scatter/Gather Status	DMA

Table 19. SIO Address Decoding (Continued)

Address	Address				Type	Name	Block
	FEDC	BA98	7654	3210			
041Dh(1)	0000	0100	0001	1101	ro	CH5 Scatter/Gather Status	DMA
041Eh(1)	0000	0100	0001	1110	ro	CH6 Scatter/Gather Status	DMA
041Fh(1)	0000	0100	0001	1111	ro	CH7 Scatter/Gather Status	DMA
0420h(1)	0000	0100	0010	00xx	r/w	CH0 Scatter/Gather Descriptor Table Pointer	DMA
0424h(1)	0000	0100	0010	01xx	r/w	CH1 Scatter/Gather Descriptor Table Pointer	DMA
0428h(1)	0000	0100	0010	10xx	r/w	CH2 Scatter/Gather Descriptor Table Pointer	DMA
042Ch(1)	0000	0100	0010	11xx	r/w	CH3 Scatter/Gather Descriptor Table Pointer	DMA
0434h(1)	0000	0100	0011	01xx	r/w	CH5 Scatter/Gather Descriptor Table Pointer	DMA
0438h(1)	0000	0100	0011	10xx	r/w	CH6 Scatter/Gather Descriptor Table Pointer	DMA
043Ch(1)	0000	0100	0011	11xx	r/w	CH7 Scatter/Gather Descriptor Table Pointer	DMA
0481h	0000	0100	1000	0001	r/w	DMA CH2 High Page Register	DMA
0482h	0000	0100	1000	0010	r/w	DMA CH3 High Page Register	DMA
0483h	0000	0100	1000	0011	r/w	DMA CH1 High Page Register	DMA
0487h	0000	0100	1000	0111	r/w	DMA CH0 High Page Register	DMA
0489h	0000	0100	1000	1001	r/w	DMA CH6 High Page Register	DMA
048Ah	0000	0100	1000	1010	r/w	DMA CH7 High Page Register	DMA
048Bh	0000	0100	1000	1011	r/w	DMA CH5 High Page Register	DMA
04D0	0000	0100	1101	0000	r/w	INT CNTRL-1 Edge Level Control Register	Control
04D1	0000	0100	1101	0001	r/w	INT CNTRL-2 Edge Level Control Register	Control
04D6h	0000	0100	1101	0010	wo	DMA2 Extended Mode Register	DMA

NOTE:
1. The I/O address of this register is relocatable. The value shown in this table is the default address location.

5.5.1.2 BIOS Memory Space

The 128 Kb BIOS memory space is located at 000E0000h to 000FFFFFh (top of 1 Mb), and is aliased at FFFE0000h to FFFFFFFFh (top of 4 Gb) and FFEE0000h to FFEFFFFFh (top of 4 Gb-1 Mb). The aliased regions account for the CPU reset vector and the uncertainty of the state of A20GATE when a software reset occurs. This 128 Kb block is split into two 64 Kb blocks. The top 64 Kb is always enabled while the bottom 64 Kb can be enabled or disabled (the aliases automatically match). Enabling the lower 64 Kb BIOS space (000E0000h to 000EFFFFh, 896 Kb-960 Kb) results in positively decoding this region and enables the BIOSCS# signal generation. The upper 64 Kb is positively decoded only if bit 6 = 1 in the ISA Clock Divisor Register. Otherwise this region is subtractively decoded. Positively decoding these cycles expedites BIOS cycles to the ISA Bus. Note that both of these regions are subtractively decoded if bit 4 in the MEMCS# Control Register is set to a 1.

When PCI master accesses to the 128 Kb BIOS space at 4 Gb-1 Mb are forwarded to the ISA Bus, the LA20 line is driven to a 1 to avoid aliasing at the 15 Mb area. The 4 Gb-1 Mb BIOS decode area accounts for the condition when A20M# is asserted and an ALT-CTRL-DEL reset is generated. The CPU's reset vector will access 4 Gb-1 Mb. When this gets forwarded to ISA, AD[32:24] are truncated and the access is aliased to 16 Mb-1 Mb = 15 Mb space. If ISA memory is present at 15 Mb, there will be contention. Forcing LA20 high aliases this region to 16 Mb. The alias here is permissible since this is the 80286 reset vector location.

In addition to the normal 128 Kb BIOS space, the SIO supports an additional 384 Kb BIOS space. The SIO can support a total of 512 Kb BIOS space. The additional 384 Kb region can only be accessed by PCI masters and resides at FFF80000h to FFFDFFFFh. When enabled via the UBCSA Register, memory accesses within this region will be positively decoded, forwarded to the ISA Bus, and encoded BIOSCS# will be generated. When forwarded to the ISA Bus, the PCI AD[23:20] signals will be propagated to the ISA LA[23:20] lines as all 1's which will result in aliasing this 512 Kb region at the top of the 16 Mb space. To avoid contention, ISA add-in memory must not be present in this space.

All PCI cycles positively decoded in the enabled BIOS space will be broadcast to the ISA Bus. Since the BIOS device is 8 or 16 bits wide and typically has very long access times, PCI burst reads from the BIOS space will invoke "disconnect target termination" semantics after the first data transaction in order to meet the PCI incremental latency guidelines.

The following tables and diagrams describe the operation of the SIO in response to PCI BIOS space accesses.

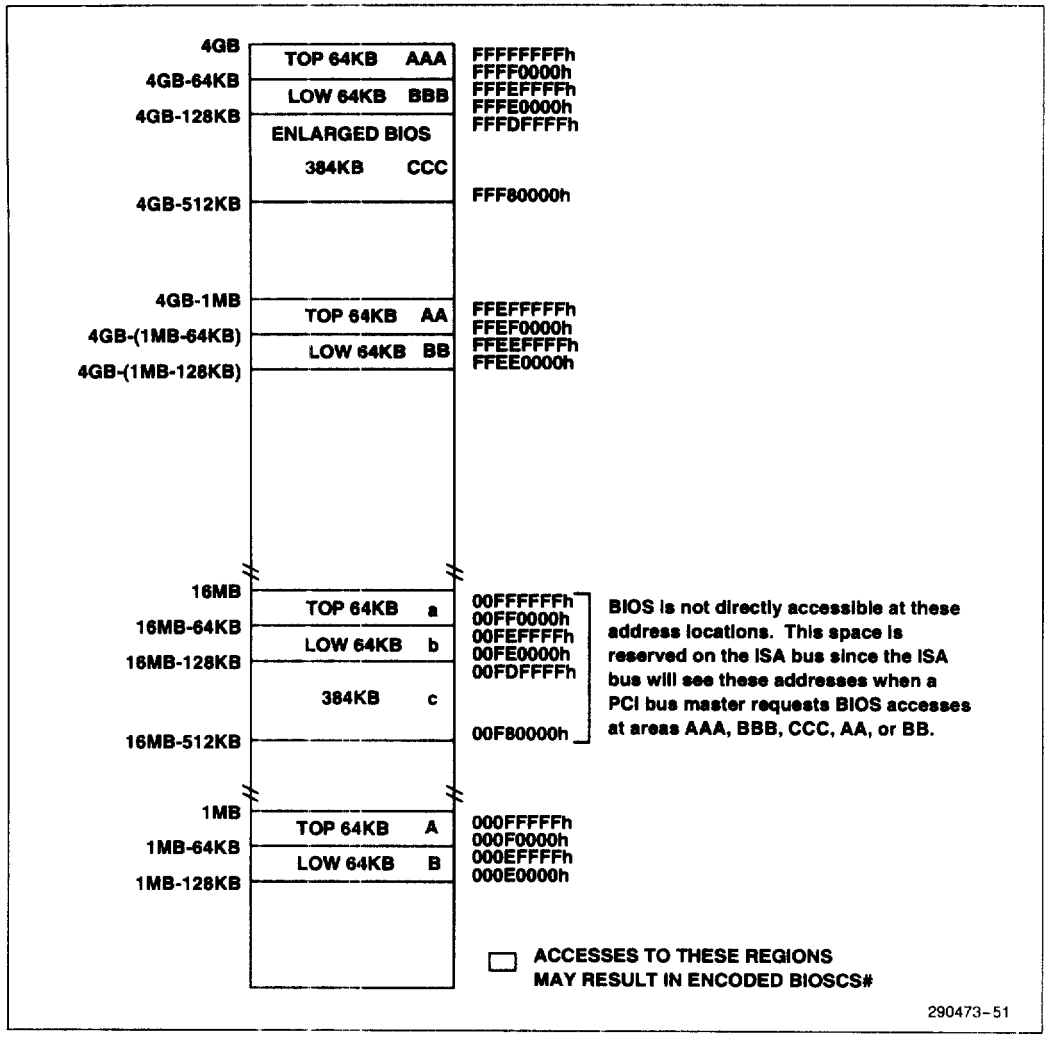


Figure 12. BIOS Space Decode Map

The BIOS space decode map, Figure 12, shows the possible BIOS spaces and the aliases throughout the memory space. The various regions are designated with code letters; "a's" for the top 64 Kb, "b's" for the low 64 Kb, and "c's" for the enlarged space.

Table 20 indicates the SIO's response to PCI BIOS space accesses based on its configuration state.

Table 20. PCI Master BIOS Space Decoding

Master	Region	Top 64 Kb BIOS Positive Decode Enabled(1)	Low 64 Kb BIOS Enabled(2)	Enlarged BIOS Enabled(3)	Encoded BIOSCS# Generated	LA20	Positive PCI Decode	Subtractive PCI Decode
PCI	A	0	x	x	Yes	Pass (0)	No	Yes
PCI	A	1	x	x	Yes	Pass (0)	Yes(5)	No(5)
PCI	B	x	0	x	No	Pass (0)	No	Yes
PCI	B	x	1	x	Yes	Pass (0)	Yes(5)	No(5)
PCI	a	1	x	x	No	Pass (1)	No	Yes
PCI	b	x	0	x	No	Pass (1)	No	Yes
PCI	c	x	x	0	No	Pass (1)	No	Yes
PCI	AA	0	x	x	Yes	1	No	Yes(4)
PCI	AA	1	x	x	Yes	1	Yes(4,5)	No(5)
PCI	BB	x	0	x	No	x	No	No
PCI	BB	x	1	x	Yes	1	Yes(4,5)	No(5)
PCI	AAA	0	x	x	Yes	Pass (1)	No	Yes(4)
PCI	AAA	1	x	x	Yes	Pass (1)	Yes(4,5)	No(5)
PCI	BBB	x	0	x	No	x	No	No
PCI	BBB	x	1	x	Yes	Pass (1)	Yes(4,5)	No(5)
PCI	CCC	x	x	0	No	x	No	No
PCI	CCC	x	x	1	Yes	Pass (1)	Yes(4)	No

NOTES:
1. The column labeled "Top 64 Kb BIOS Positive Decode Enable" shows the value of the ISA Clock Register bit 6. This bit determines the decoding for memory regions A, AA, and AAA (1 = positive, 0 = negative decoding).
   Note that if bit 4 in the MEMCS# Control Register is set to a 1 (Global MEMCS# decode enabled), the positive decoding function enabled by having ISA Clock Register bit 6 = 1 is ignored. Subtractive decoding is provided, instead.
2. The column labeled "Low 64 Kb BIOS Enable" shows the value of the Utility Bus Chip Select Enable A Bit 6. This bit determines whether memory regions B, BB, and BBB are enabled (bit = 1) or disabled (bit = 0).
3. The column labeled "Enlarged BIOS Enabled" shows the value of the Utility Bus Chip Select Enable A Bit 7. This bit determines whether memory region CCC is enabled (bit = 1) or disabled (bit = 0).
4. ISA memory is not allowed to be enabled at the corresponding aliased areas or contention will result.
5. When bit 4 in the MEMCS# Control Register is set to a 1 (Global MEMCS# decode enabled), positive decoding for these areas will be disabled. The SIO will only provide subtractive decoding in this case.

5.5.1.3 MEMCS# Decoding

For MEMCS# decoding, the SIO decodes sixteen ranges. Fourteen of these ranges can be enabled or disabled independently for both read and write cycles. The fifteenth range (0 KB–512 KB) and sixteenth range (programmable from 1 MB up to 512 MB in 2 MB increments) can be enabled or disabled only. Addresses within these enabled regions generate a MEMCS# signal that can be used by the host bridge to know when to forward PCI cycles to main memory. A seventeenth range is available that can be used to identify a "memory hole". Addresses within this hole will not generate a MEMCS#. The address regions are summarized:

- 0 KB to 512 KB Memory (can only be disabled if MEMCS# is completely disabled)
- 512 KB to 640 KB Memory
- (1 MB–64 KB) to 1 MB Memory (BIOS Area)
- 768 KB to 918 KB in 16 KB sections (total of 8 sections)
- 918 KB to 983 KB in 16 KB sections (total of 4 sections)
- 1M-to-programmable boundary on 2 MB increments from 2 MB up to 512 MB
- programmable "memory hole" in 64 KB increments between 1 MB and 16 MB

Table 21 and Figure 13 show the registers and decode areas for MEMCS#.

Table 21. MEMCS# Decoding Register Summary

MAR Registers	Attribute	Memory Segments	Comments
MCSCON[4] = 0	Disable	Disable MEMCS# Function	Enable/Disable MEMCS# Function
MCSCON[4] = 1	Enable	Enable MEMCS# Function	When Enabled, 0 KB to 512 KB Range is also Automatically Enabled (RE/WE)
MCSTOH/ MCSBOH	MEMCS# Hole	100000h–0FFFFFFh	1 MB to 16 MB Hole in MEMCS# Region
MCSTOM	MEMCS# Top	200000h–1FFFFFFFh	2 MB to 512 MB Top of MEMCS# Region
MCSCON[1:0]	[0] = RE[1] = WE	080000h–09FFFFh	512 KB to 640 KB R/W Enable
MCSCON[3:2]	[2] = RE[3] = WE	0F0000h–0FFFFFh	BIOS Area R/W Enable
MAR1[1:0]	[0] = RE[1] = WE	0C0000h–0C3FFFh	ISA Add-On BIOS R/W Enable
MAR1[3:2]	[2] = RE[3] = WE	0C4000h–0C7FFFh	ISA Add-On BIOS R/W Enable
MAR1[5:4]	[4] = RE[5] = WE	0C8000h–0CBFFFh	ISA Add-On BIOS R/W Enable
MAR1[7:6]	[6] = RE[7] = WE	0CC000h–0CFFFFh	ISA Add-On BIOS R/W Enable
MAR2[1:0]	[0] = RE[1] = WE	0D0000h–0D3FFFh	ISA Add-On BIOS R/W Enable
MAR2[3:2]	[2] = RE[3] = WE	0D4000h–0D7FFFh	ISA Add-On BIOS R/W Enable
MAR2[5:4]	[4] = RE[5] = WE	0D8000h–0DBFFFh	ISA Add-On BIOS R/W Enable
MAR2[7:6]	[6] = RE[7] = WE	0DC000h–0DFFFFh	ISA Add-On BIOS R/W Enable
MAR3[1:0]	[0] = RE[1] = WE	0E0000h–0E3FFFh	BIOS Extension R/W Enable
MAR3[3:2]	[2] = RE[3] = WE	0E4000h–0E7FFFh	BIOS Extension R/W Enable
MAR3[5:4]	[4] = RE[5] = WE	0E8000h–0EBFFFh	BIOS Extension R/W Enable
MAR3[7:6]	[6] = RE[7] = WE	0EC000h–0EFFFFh	BIOS Extension R/W Enable

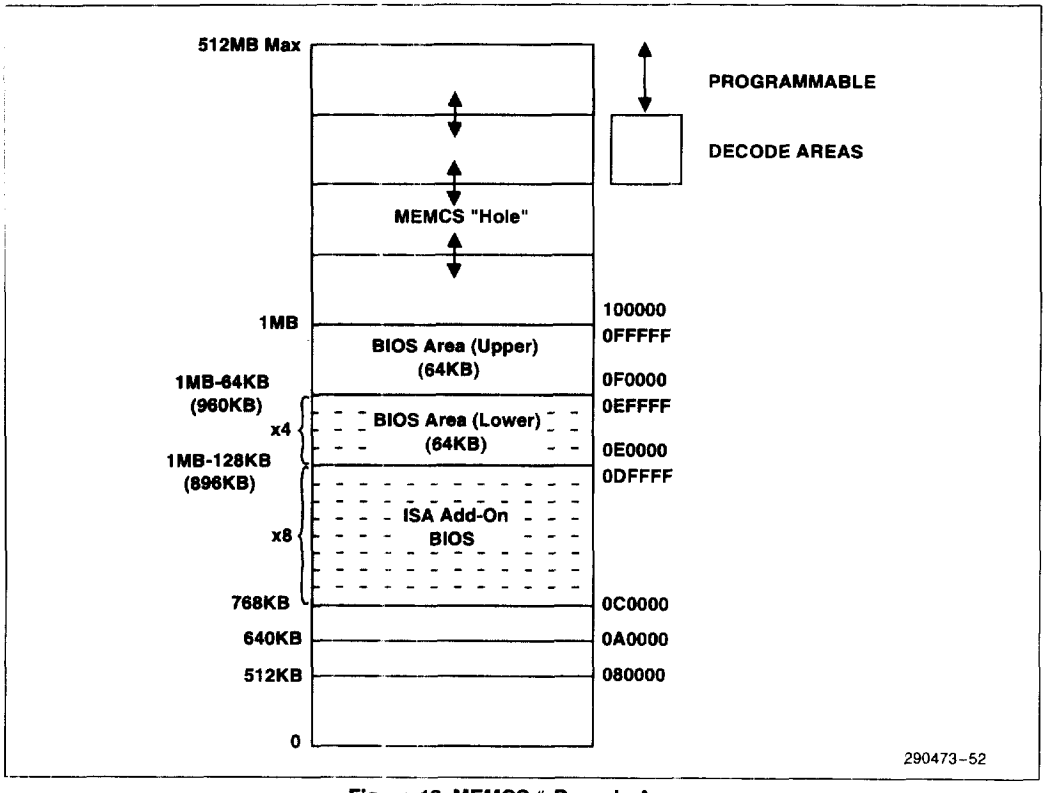


Figure 13. MEMCS# Decode Areas

The SIO generates MEMCS# from the PCI address. MEMCS# is generated from the clock edge after FRAME# is sampled active. MEMCS# will only go active for one PCI clock period. The SIO does not take any other action as a result of this decode other than generating MEMCS#. It is the responsibility of the device using the MEMCS# signal to generate DEVSEL#, TRDY# and any other cycle response. The device using MEMCS# will always generate DEVSEL# on the next clock. This fact can be used to avoid an extra clock delay in the subtractive decoder described in the next section.

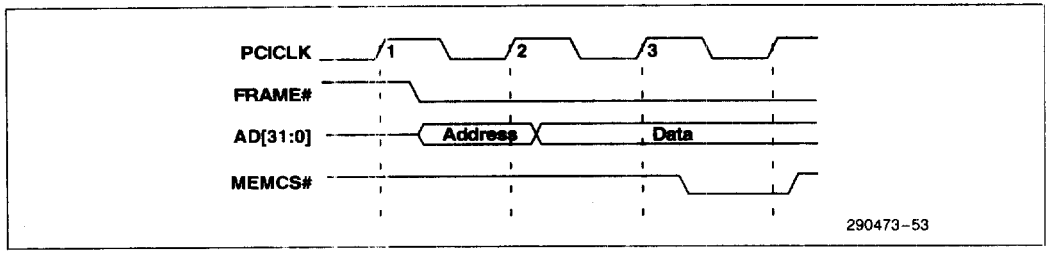


Figure 14. MEMCS# Generation

5.5.1.4 Subtractively Decoded Cycles to ISA

The addresses that reside on the ISA Bus could be highly fragmented. For this reason, subtractive decoding is used to forward PCI cycles to the ISA Bus. An inactive DEVSEL# will cause the SIO to forward the PCI cycle to the ISA Bus. The DEVSEL# sample point can be configured for three different settings. If the "fast" point is selected, the cycle will be forwarded to ISA when DEVSEL# is inactive at the F sample point as shown in Figure 15. If the "typical" point is selected, DEVSEL# will be sampled on both F and T, and if inactive, will be forwarded to the ISA Bus. Likewise, if the "slow" point is selected, DEVSEL# will be sampled at F, T and S. The sample point should be configured to match the slowest PCI device in the system. This capability reduces the latency to ISA slaves when all PCI devices are "fast" and also allows for devices with slow decoding. Note that when these unclaimed cycles are forwarded to the ISA Bus, the SIO will drive the DEVSEL# active.

Since an active MEMCS# will always result in an active DEVSEL# at the "Slow" sample point. MEMCS# is used as an early indication of DEVSEL#. In this case, if the device using MEMCS# is the only "slow" agent in the system, the sample point can be moved in to the "typical" edge.

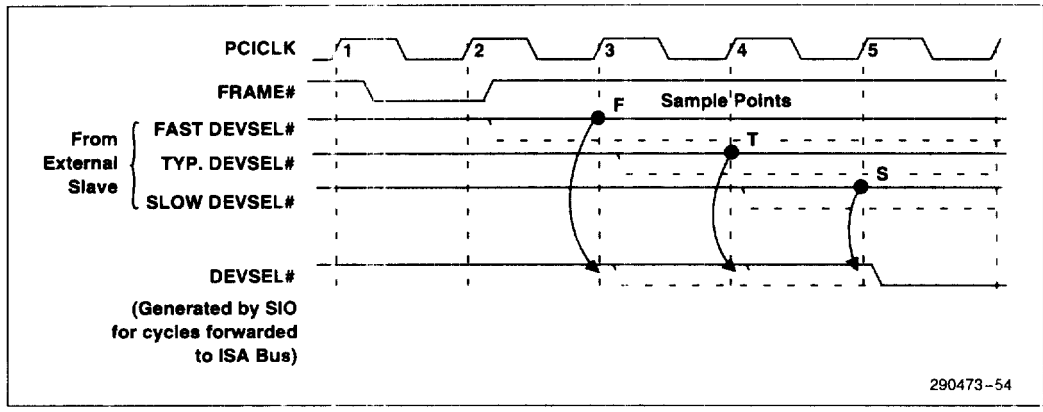


Figure 15. DEVSEL# Generation

Unclaimed PCI cycles with memory addresses above 16M and I/O addresses above 64K will not be forwarded to the ISA Bus. The SIO will not respond with DEVSEL# (BIOS accesses are an exception to this). This is required to avoid the possibility of aliasing. Under this condition, these unclaimed cycles will be recognized as such by the originating master and the master will use "master-abort" semantics to terminate the PCI cycle.

5.5.2 DMA/ISA MASTER CYCLE ADDRESS DECODER

The SIO also contains a decoder which is used to determine the destination of ISA master and DMA master cycles. This decoder provides:

Positive Decode to PCI: Positively decodes addresses to be forwarded to the PCI Bus. This includes addresses residing directly on PCI as well as addresses that reside on the back side of PCI bridges (Host Bridges).

Access to SIO Internal Registers: Positively decodes addresses to registers within the SIO

BIOS Accesses: Positively decodes BIOS memory accesses and generates encoded BIOSCS#.

Utility Bus Chip Selects: Positively decodes utility bus chip selects.

Subtractive Decode: Subtractively decodes cycles to be contained to the ISA Bus.

5.5.2.1 Positive Decode to PCI

ISA master or DMA addresses that are positively decoded by this decoder will be propagated to the PCI Bus. This is the only way to forward a cycle from an ISA master or the DMA to the PCI Bus. If the cycle is not decoded by this decoder it will *not* be forwarded to the PCI Bus.

This decoder has several memory address regions to positively decode cycles that should be forwarded to the PCI Bus. These regions are listed below. Regions "a" through "e" are fixed and can be enabled or disabled independently. Region "f" defines a space starting at 1M with a programmable upper boundary up to 16 MB. Within this region a hole can be opened. Its size and location are programmable to allow a hole to be opened in the memory space. A memory address above 16 MB will be forwarded to the PCI Bus automatically. This is possible only during DMA cycles in which the DMA has been programmed for 32-bit addressing above 16 MB.

a. Memory: 0 KB–512 KB
b. Memory: 512 KB–640 KB
c. Memory: 640 KB–768 KB (Video buffer)
d. Memory: 768 KB–896 KB in eight 16K sections (Expansion ROM)
e. Memory: 896 KB–960 KB (lower BIOS area)
f. Memory: 1 MB-to-X MB (up to 16 MB) within which a hole can be opened. Accesses to the hole are not forwarded to PCI. The top of the region can be programmed on 64 KByte boundaries up to 16 MB. The hole can be between 64 KB and 8 MB in size in 64 KB increments located on any 64 KB boundary. (Refer to the ISA Address Decoder Register in the register description section, Section 5.5.2)
g. Memory: > 16 MB automatically forwarded to PCI

Figure 16 shows a map of the ISA master/DMA decode regions and Table 22 summarizes the registers used to configure the decoder.

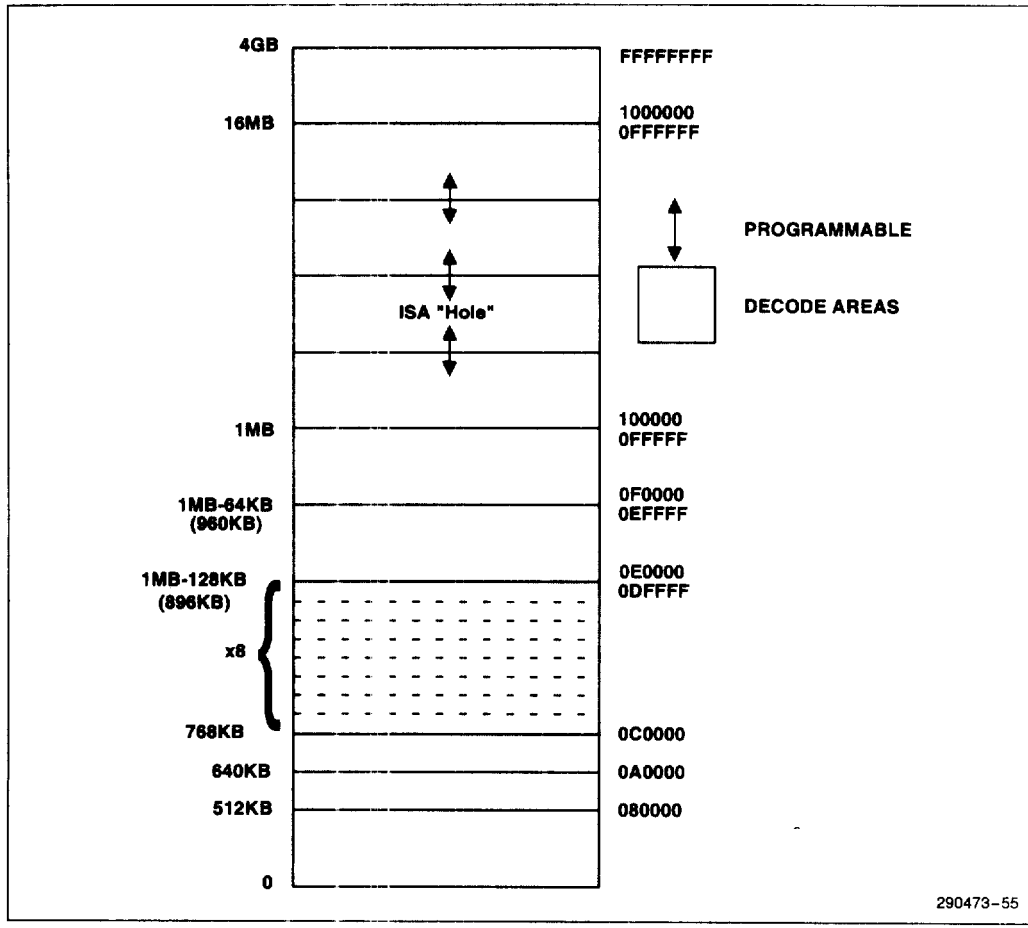


Figure 16. ISA Master/DMA to PCI Bus Decoder Regions

Table 22. ISA Master/DMA to PCI Bus Decoding Register Summary

MAR Registers	Attribute	Memory Segments	Comments
IADCON[7:4]	ISA Memory Top	100000h–0FFFFFFh	1 MB to 16 MB Top of ISA Region
IADTOH/IADBOH	ISA Hole	100000h–0FFFFFFh	1 MB to 16 MB Hole in ISA Region
IADCON[0]	Enable/Disable	000000h–07FFFFh	0 to 512 KB Enable/Disable
IADCON[1]	Enable/Disable	080000h–09FFFFh	512 KB to 640 KB Enable/Disable
IADCON[2]	Enable/Disable	0A0000h–0BFFFFh	640 KB to 768 KB Enable/Disable
IADCON[3]*	Enable/Disable	0E0000h–0EFFFFh	896 KB to 960 KB Lower BIOS Enable/Disable
IADRBE[0]	Enable/Disable	0C0000h–0C3FFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[1]	Enable/Disable	0C4000h–0C7FFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[2]	Enable/Disable	0C8000h–0CBFFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[3]	Enable/Disable	0CC000h–0CFFFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[4]	Enable/Disable	0D0000h–0D3FFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[5]	Enable/Disable	0D4000h–0D7FFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[6]	Enable/Disable	0D8000h–0DBFFFh	ISA Add-On BIOS (Expansion ROM) Enable
IADRBE[7]	Enable/Disable	0DC000h–0DFFFFh	ISA Add-On BIOS (Expansion ROM) Enable

NOTE:
* This can be overridden by bit 6 of the UBCSA Register being set to a 1.

5.5.2.2 SIO Internal Registers

Most of the internal SIO registers are accessible by ISA masters. Table 19 lists the registers that are not accessible by ISA masters. Registers accessed by ISA masters are run as 8-bit extended I/O cycles.

5.5.2.3 BIOS Accesses

The 128K BIOS memory space is located at 000E0000h to 000FFFFFh, and is aliased at FFFE0000h to FFFFFFFFh (top of 4 GB) and FFEE0000h to FFEFFFFFh (top of 4 GB–1 MB). The aliased regions account for the CPU reset vector and the uncertainty of the state of A20GATE when a software reset occurs. This 128K block is split into two 64K blocks. The top 64K is always enabled while the bottom 64K can be enabled or disabled (the aliases automatically match). ISA masters can only access BIOS in the 000E0000 to 000FFFFFh region.

ISA originated accesses to the enabled 64K sections of the BIOS space (000E0000h–000FFFFFh) will activate the encoded BIOSCS# signal. ISA originated cycles will not be forwarded to the PCI Bus. Encoded BIOSCS# is combinatorially generated from the ISA, SA, and LA address bus. Encoded BIOSCS# is disabled during refresh and DMA cycles. The ISA Master/DMA BIOS Decoding Table indicates the SIO's response to BIOS accesses based on the configuration state.

Table 23. ISA Master/DMA BIOS Decoding

Cycle		SIO Configuration			SIO Response		
Master	**Region(1)**	**Top 64 KB PCI Positive Decode Enabled(2)**	**Low 64 KB BIOS Enabled(3)**	**Forward Low 64 KB to PCI Enabled(4)**	**Encoded BIOSCS# Generated**	**Forward to PCI**	**Contain to ISA**
ISA/DMA	A	x	x	x	Yes	No	Yes
ISA/DMA	B	x	0(5)	0	No	No	Yes
ISA/DMA	B	x	0(5)	1	No	Yes	No
ISA/DMA	B	x	1	x	Yes	No	Yes
ISA/DMA	a	These cycles will be forwarded to PCI dependent on the state of the ISA Address Decoder Configuration Registers. Encoded BIOSCS# will not be generated for any of these cycles.					
ISA/DMA	b						
ISA/DMA	c						

NOTES:
1. The memory sections referenced can be found in Figure 12.
2. The column labeled "Top 64 KB BIOS Positive Decode Enabled" shows the value of the ISA Clock Divisor Configuration Register bit 6. This bit determines how the memory region is decoded (0 = subtractively decoded, 1 = positively decoded).
3. The column labeled "Low 64 KB BIOS Enable" shows the value of the Utility Bus Chip Select Enable A Configuration Register bit 6. This bit determines if the memory region is enabled (bit = 1) or disabled (bit = 0).
4. The column labeled "Forward Low 64 KB to PCI Enables" shows the value of the ISA Address Decoder Control Configuration Register Bit 3. This bit determines whether PCI Bus forwarding is enabled (bit = 1) or disabled (bit = 0).
5. Forward to PCI if IADCON Bit 6 = 1.

5.5.2.4 Utility Bus Encoded Chip Selects

The SIO generates encoded chip selects for certain functions that are located on the utility bus (formerly X-Bus). The encoded chip selects are generated combinatorially from the ISA SA[15:0] address bus. The encoded chip selects are decoded externally (see Figure 19).

The encoded chip select table (Table 24) shows the addresses that result in encoded chip select generation. Chip selects can be enabled or disabled via configuration registers. In general, the addresses shown in Table 24 do not reside in the SIO itself. Write only addresses 70h, 372h, 3F2h are exceptions since particular bits from these registers reside in the SIO. For ISA master cycles, the SIO will respond to writes to address 70h, 372h, and 3F2h by generating IOCHRDY and writing to the appropriate bits.

Note that the SIO monitors read accesses to address 60h to support the mouse function. In this case, IOCHRDY is not generated.

Table 24. Encoded Chip Select Table

Address	Address				Type	Name	Encoded Chip Select
	FEDC	**BA98**	**7654**	**3210**			
0060h	0000	0000	0110	00x0	r/w	Keyboard Controller	KEYBRDCS#
0064h	0000	0000	0110	01x0	r/w	Keyboard Controller	KEYBRDCS#
0070h	0000	0000	0111	0xx0	w	Real Time Clock Address	RTCALE#

Table 24. Encoded Chip Select Table (Continued)

Address	Address				Type	Name	Encoded Chip Select
	FEDC	BA98	7654	3210			
0071h	0000	0000	0111	0xx1	r/w	Real Time Clock Data	RTCCS#
0170h	0000	0001	0111	0000	r/w	Secondary Data Register	IDECS0#
0171h	0000	0001	0111	0001	r/w	Secondary Error Register	IDECS0#
0172h	0000	0001	0111	0010	r/w	Secondary Sector Count Register	IDECS0#
0173h	0000	0001	0111	0011	r/w	Secondary Sector Number Register	IDECS0#
0174h	0000	0001	0111	0100	r/w	Secondary Cylinder Low Register	IDECS0#
0175h	0000	0001	0111	0101	r/w	Secondary Cylinder High Register	IDECS0#
0176h	0000	0001	0111	0110	r/w	Secondary Drive/Head Register	IDECS0#
0177h	0000	0001	0111	0111	r/w	Secondary Status Register	IDECS0#
01F0h	0000	0001	1111	0000	r/w	Primary Data Register	IDECS0#
01F1h	0000	0001	1111	0001	r/w	Primary Error Register	IDECS0#
01F2h	0000	0001	1111	0010	r/w	Primary Sector Count Register	IDECS0#
01F3h	0000	0001	1111	0011	r/w	Primary Sector Number Register	IDECS0#
01F4h	0000	0001	1111	0100	r/w	Primary Cylinder Low Register	IDECS0#
01F5h	0000	0001	1111	0101	r/w	Primary Cylinder High Register	IDECS0#
01F6h	0000	0001	1111	0110	r/w	Primary Drive/Head Register	IDECS0#
01F7h	0000	0001	1111	0111	r/w	Primary Status Register	IDECS0#
0278h	0000	0010	0111	1x00	r/w	LPT3 PP Data Latch	LPTCS#
0279h	0000	0010	0111	1x01	r	LPT3 PP Status	LPTCS#
027Ah	0000	0010	0111	1x10	r/w	LPT3 PP Control	LPTCS#
027Bh	0000	0010	0111	1x11	r/w		LPTCS#
02F8h	0000	0010	1111	1000	r/w	COM2 SP Transmit/Receive Register	COM2CS#
02F9h	0000	0010	1111	1001	r/w	COM2 SP Interrupt Enable Register	COM2CS#
02FAh	0000	0010	1111	1010	r	COM2 SP Interrupt Identification Register	COM2CS#
02FBh	0000	0010	1111	1011	r/w	COM2 SP Line Control Register	COM2CS#
02FCh	0000	0010	1111	1100	r/w	COM2 SP Modem Control Register	COM2CS#
02FDh	0000	0010	1111	1101	r	COM2 SP Line Status Register	COM2CS#
02FEh	0000	0010	1111	1110	r	COM2 SP Modem Status Register	COM2CS#
02FFh	0000	0010	1111	1111	r/w	COM2 SP Scratch Register	COM2CS#

Table 24. Encoded Chip Select Table (Continued)

Address	Address				Type	Name	Encoded Chip Select
	FEDC	BA98	7654	3210			
0370h	0000	0011	0111	0000	r/w	Secondary Floppy Disk Extended Mode Register	FLOPPYCS#
0371h	0000	0011	0111	0001	r/w	Secondary Floppy Disk Extended Mode Register	FLOPPYCS#
0372	0000	0011	0111	0010	w	Secondary Floppy Disk Digital Output Register	FLOPPYCS#
0373h	0000	0011	0111	0011	r/w	Reserved	FLOPPYCS#
0374h	0000	0011	0111	0100	r/w	Secondary Floppy Disk Status Register	FLOPPYCS#
0375h	0000	0011	0111	0101	r/w	Secondary Floppy Disk Data Register	FLOPPYCS#
0376h	0000	0011	0111	0110	r/w	Secondary Alternate Status Register	IDECS1#
0377h	0000	0011	0111	0111	r	Secondary Drive Address Register	IDECS1#
0377h*	0000	0011	0111	011x	r/w	Secondary Floppy Disk Digital Input Register	FLOPPYCS#
0378h	0000	0011	0111	1x00	r/w	LPT2 PP Data Latch	LPTCS#
0379h	0000	0011	0111	1x01	r	LPT2 PP Status	LPTCS#
037Ah	0000	0011	0111	1x10	r/w	LPT2 PP Control	LPTCS#
037Bh	0000	0011	0111	1x11	r/w		LPTCS#
03BCh	0000	0011	1011	1100	r/w	LPT1 PP Data Latch	LPTCS#
03BDh	0000	0011	1011	1101	r	LPT1 PP Status	LPTCS#
03BEh	0000	0011	1011	1110	r/w	LPT1 PP Control	LPTCS#
03BFh	0000	0011	1011	1111	r/w		LPTCS#
03F0h	0000	0011	1111	0000	r/w	Primary Floppy Disk Extended Mode Register	FLOPPYCS#
03F1h	0000	0011	1111	0001	r/w	Primary Floppy Disk Extended Mode Register	FLOPPYCS#
03F2h	0000	0011	1111	0010	w	Primary Floppy Disk Digital Output Register	FLOPPYCS#
03F3h	0000	0011	1111	0011	r/w	Reserved	FLOPPYCS#
03F4h	0000	0011	1111	0100	r/w	Primary Floppy Disk Status Register	FLOPPYCS#
03F5h	0000	0011	1111	0101	r/w	Primary Floppy Disk Data Register	FLOPPYCS#
03F6h	0000	0011	1111	0110	r/w	Primary Drive Alternate Status Register	IDECS1#
03F7h	0000	0011	1111	0111	r	Primary Drive Address Register	IDECS1#
03F7h*	0000	0011	1111	011x	r/w	Primary Floppy Disk Digital Input Register	FLOPPYCS#

Table 24. Encoded Chip Select Table (Continued)

Address	Address				Type	Name	Encoded Chip Select
	FEDC	BA98	7654	3210			
03F8h	0000	0011	1111	1000	r/w	COM1 SP Transmit/Receive Register	COM1CS#
03F9h	0000	0011	1111	1001	r/w	COM1 SP Interrupt Enable Register	COM1CS#
03FAh	0000	0011	1111	1010	r	COM1 SP Interrupt Identification Register	COM1CS#
03FBh	0000	0011	1111	1011	r/w	COM1 SP Line Control Register	COM1CS#
03FCh	0000	0011	1111	1100	r/w	COM1 SP Modem Control Register	COM1CS#
03FDh	0000	0011	1111	1101	r	COM1 SP Line Status Register	COM1CS#
03FEh	0000	0011	1111	1110	r	COM1 SP Modem Status Register	COM1CS#
03FFh	0000	0011	1111	1111	r/w	COM1 SP Scratch Register	COM1CS#
0800h–08FFh	0000	1000	xxxx	xxxx	r/w		CFIGMEMCS#
0C00h	0000	1100	0000	0000	r/w		CPAGECS#

NOTE:
*If both the IDE and Floppy Drive are located on the UBUS, FLOPPYCS# will not be generated, IDECS1# will be generated.

5.5.2.5 Subtractive Decode to ISA

ISA master and DMA cycles not positively decoded by the ISA decoder are contained to the ISA Bus.

5.6 Data Buffering

The SIO contains data buffers to isolate the PCI Bus from the ISA Bus. The buffering is described from two perspectives: PCI master accesses to the ISA Bus (Posted Write Buffer) and DMA/ISA master accesses to the PCI Bus (Line Buffer). Temporarily buffering the data requires buffer management logic to ensure that the data buffers remain coherent.

5.6.1 DMA/ISA MASTER LINE BUFFER

An 8-byte Line Buffer is used to isolate the ISA Bus's slower I/O devices from the PCI Bus. The Line Buffer is bi-directional and is used by ISA masters and the DMA controller to assemble and disassemble data. Only memory data written to or read from the PCI Bus by an ISA master or DMA is assembled/disassembled using this 8 byte line buffer. I/O cycles do not use the buffer.

Bits 0 and 1 of the PCI Control Register set the buffer to operate in either single transaction mode (bit = 0) or 8-byte mode (bit = 1). Note that ISA masters and DMA controllers can have their buffer modes configured separately.

In single transaction mode, the buffer will store only one transaction. For DMA/ISA master writes, this single transaction buffer looks like a posted write buffer. As soon as the ISA cycle is complete, a PCI cycle is scheduled. Subsequent DMA/ISA master writes are held off in wait-states until the buffer is empty. For DMA/ISA master reads, only the data requested is read over the PCI Bus. For instance, if the DMA channel is programmed in 16-bit mode, 16 bits of data will be read from PCI. As soon as the requested data is valid on the PCI bus, it is latched into the Line Buffer and the ISA cycle is then completed, as timing allows. Single transaction mode will guarantee strong read and write ordering through the buffers.

In 8 byte mode, for write data assembly, the Line Buffer acts as two individual 4 byte buffers working in ping pong fashion. For read data disassembly, the Line Buffer acts as one 8 byte buffer.

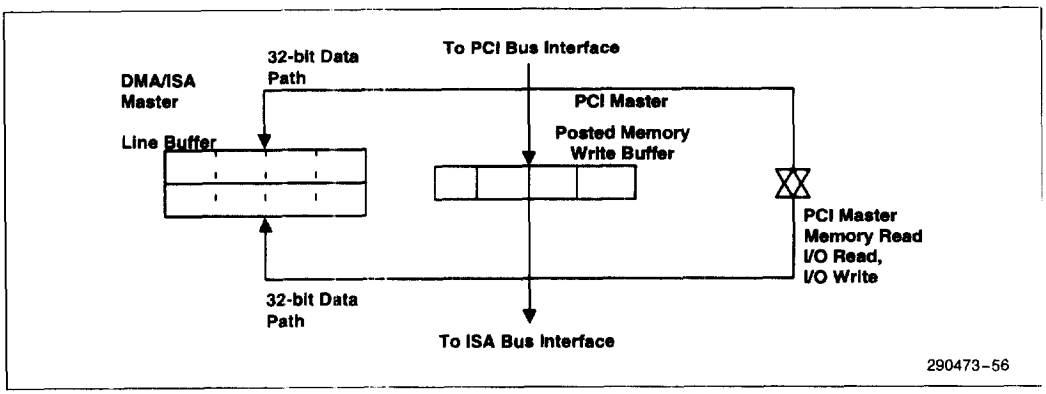


Figure 17. SIO Buffer Diagram

5.6.2 PCI MASTER POSTED WRITE BUFFER

PCI master memory write cycles destined to ISA memory are buffered in a 32-bit Posted Write Buffer. The PCI Memory Write and Memory Write and Invalidate commands are all treated as a memory write and can be posted, subject to the Posted Write Buffer status. The Posted Write Buffer has an address associated with it. A PCI master memory write can be posted any time the posted write buffer is empty and write posting is enabled (bit 2 of the PCI Control Configuration Register is set to a 1). Also, the ISA Bus must not be occupied. If the posted write buffer contains data, the PCI master write cycle is retried. If the posted write buffer is disabled, the SIO's response to a PCI master memory write is dependent on the state of the ISA Bus. If the ISA Bus is available and the posted write buffer is disabled, the cycle will immediately be forwarded to the ISA Bus (TRDY# will not be asserted until the ISA cycle has completed). If the ISA Bus is busy and the posted write buffer is disabled, the cycle is retried.

Memory read and I/O read and I/O write cycles do not use the 32-bit Posted Write Buffer.

5.6.3 BUFFER MANAGEMENT

Any time data is temporarily stored in the buffers between the ISA Bus and the PCI Bus, there are potential data coherency problems.

The SIO contains buffer management circuitry which guarantees data coherency by intercepting synchronization protocol between the buses and managing the buffers before synchronization communication between the buses is complete. The buffers are flushed or invalidated as appropriate before a bus cycle is allowed to occur in cases where data coherency could be lost.

5.6.3.1 DMA/ISA Master Line Buffer—Write State

When the DMA/ISA Master Line Buffer contains data that is to be written to the PCI Bus, it is in the Write State. The 8-byte line buffer is flushed when the line becomes full, when a subsequent write is a line miss, when a subsequent write would overwrite an already valid byte, or when a subsequent cycle is a read. The ISA master or DMA cycle that triggers the buffer flush will be held in wait-states until the flush is complete. The buffer is also flushed whenever there is a change in ISA Bus ownership as indicated by any DACK# signal going inactive.

Once the buffer is scheduled to be flushed to PCI, any PCI cycle to the SIO or ISA Bus will get retried by the SIO.

5.6.3.2 DMA/ISA Master Line Buffer—Read State

When the DMA/ISA Master Line Buffer contains data that has been read from the PCI Bus, it is in the Read State. The data in the buffer will be invalidated when the SIO accepts a PCI memory or I/O write cycle. The line buffer in the read state is also invalidated when a subsequent read is a line miss, or when a subsequent cycle is a write. The line buffer in the read state is not invalidated on a change of ISA ownership. Note that as bytes are disassembled from the line buffer, they are invalidated so that subsequent reads to the same byte will cause a line buffer miss.

5.6.3.3 PCI Master Posted Write Buffer

As soon as a PCI master has posted a memory write into the posted write buffer, the buffer is scheduled to be written to the ISA Bus. Any subsequent PCI master cycles to the SIO (including ISA Bus) will be retried until the posted write buffer is empty.

Prior to granting the ISA Bus to an ISA master or the DMA, the PCI master posted memory write buffer is flushed. Also, as long as the ISA master or DMA owns the ISA Bus, the posted write buffer is disabled. A PCI master write can not be posted while an ISA master or the DMA owns the ISA Bus.

5.7 SIO Timers

5.7.1 INTERVAL TIMERS

The SIO contains three counters that are equivalent to those found in the 82C54 programmable interval timer. The three counters are contained in one SIO timer unit, referred to as Timer-1. Each counter output provides a key system function. Counter 0 is connected to interrupt controller IRQ0 and provides a system timer interrupt for a time-of-day, diskette time-out, or other system timing functions. Counter 1 generates a refresh request signal and Counter 2 generates the tone for the speaker. Note that the 14.31818 MHz counters use OSC for a clock source.

Full details of this counter can be found in the 82C54 data sheet.

Table 25. Interval Timer Functions Table

Interval Timer Functions	
Function	**Counter 0—System Timer**
Gate	Always On
Clock In	1.193 MHz (OSC/12)
Out	INT-1 IRQ0
Function	**Counter 1—Refresh Request**
Gate	Always On
Clock In	1.193 MHz (OSC/12)
Out	Refresh Request
Function	**Counter 2—Speaker Tone**
Gate	Programmable-Port 61h
Clock In	1.193 MHz (OSC/12)
Out	Speaker

5.7.1.1 Interval Timer Address Map

Table 26 shows the I/O address map of the interval timer counters.

Table 26. Interval Timer Counters I/O Address Map

I/O Address	Register Description
040h	System Timer (Counter 0)
041h	Refresh Request (Counter 1)
042h	Speaker Tone (Counter 2)
043h	Control Word Register

Counter 0, System Timer

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

Counter 1, Refresh Request Signal

This counter provides the refresh request signal and is typically programmed for Mode 2 operation. The counter negates refresh request for one counter period (833 ns) during each count cycle. The initial count value is loaded one counter period after being written to the counter I/O address. The counter initially asserts refresh request, and negates it for 1 counter period when the count value reaches 1. The counter then asserts refresh request and continues counting from the initial count value.

Counter 2, Speaker Tone

This counter provides the speaker tone and is typically programmed for Mode 3 operation. The counter provides a speaker frequency equal to the counter clock frequency (1.193 MHz) divided by the initial count value. The speaker must be enabled by a write to port 061h (see Section 4.5.1 on the NMI Status and Control Register).

5.7.2 BIOS TIMER

5.7.2.1 Overview

The SIO provides a system BIOS Timer that decrements at each edge of its 1.04 MHz clock (derived by dividing the 8.33 MHz SYSCLK by 8). Since the state of the counter is undefined at power-up, it must be programmed before it can be used. Accesses to the BIOS Timer are enabled and disabled through the BIOS Timer Base Address Register. The timer continues to count even if accesses are disabled.

A BIOS Timer Register is provided to start the timer counter by writing an initial clock value. The BICS Timer Register can be accessed as a single 16-bit I/O port or as a 32-bit port with the upper 16-bits being "don't care" (reserved). It is up to the software to access the I/O register in the most convenient way. The I/O address of the BIOS Timer Register is software relocatable. The I/O address is determined by the value programmed into the BIOS Timer Base Address Register.

The BIOS Timer clock has a value of 1.04 MHz using an 8.33 MHz SYSCLK input (an 8 to 1 ratio will always exist between SYSCLK and the timer clock). This allows the counting of time intervals from 0 ms to approximately 65 ms. Because of the PCI clock rate, it is possible to start the counter and read the value back in less than 1 μs. The expected value of the expired interval is 0, but depending on the state of the internal clock divisor, the BIOS Timer might indicate that 1 ms has expired. Therefore, accuracy of the counter is $\pm$ 1 μs.

5.7.2.2 BIOS Timer Operations

A write operation to the BIOS Timer Register will initiate the counting sequence. The timer can be initiated by writing either the 16-bit data portion or the whole 32-bit register (upper 16 bits are "don't care"). After initialization, the BIOS timer will start decrementing until it reaches zero. Then it will stop decrementing (and hold a zero value) until initialized again.

After the timer is initialized, the current value can be read at any time and the timer can be reprogrammed (new initial value written), even before it reaches zero.

All write and read operations to the BIOS timer Register should include all 16 counter bits. Separate accesses to the individual bytes of the counter must be avoided since this can cause unexpected results (wrong count intervals).

5.8 Interrupt Controller

The SIO provides an ISA compatible interrupt controller which incorporates the functionality of two 82C59 interrupt controllers. The two controllers are cascaded so that 14 external and two internal interrupts are possible. The master interrupt controller provides IRQ[7:0] and the slave interrupt controller provides IRQ [15:8] (see Figure 18). The two internal interrupts are used for internal functions only and are not available to the user. IRQ2 is used to cascade the two controllers together and IRQ0 is used as a system timer interrupt and is tied to Interval Timer 1, Counter 0. The remaining 14 interrupt lines (IRQ1, IRQ3–IRQ15) are available for external system interrupts. Edge or level sense selection is programmable on a by-controller basis.

The Interrupt Controller consists of two separate 82C59 cores. Interrupt Controller 1 (CNTRL-1) and Interrupt Controller 2 (CNTRL-2) are initialized separately and can be programmed to operate in different modes. The default settings are: 80x86 Mode, Edge Sensitive (IRQ0-15) Detection, Normal EOI, Non-Buffered Mode, Special Fully Nested Mode disabled, and Cascade Mode. CNTRL-1 is connected as the Master Interrupt Controller and CNTRL-2 is connected as the Slave Interrupt Controller.

Note that IRQ13 is generated internally (as part of the coprocessor error support) by the SIO when bit 5 in the ISA Clock Divisor Register is set to a 1. When this bit is set to a 0, then the FERR#/IRQ13 signal is used as an external IRQ13 signal and has the same functionality as the normal IRQ13 signal. IRQ12/M is generated internally (as part of the mouse support) by the SIO when bit 4 in the ISA Clock Divisor Register is set to a 1. When set to a 0, the standard IRQ12 function is provided.

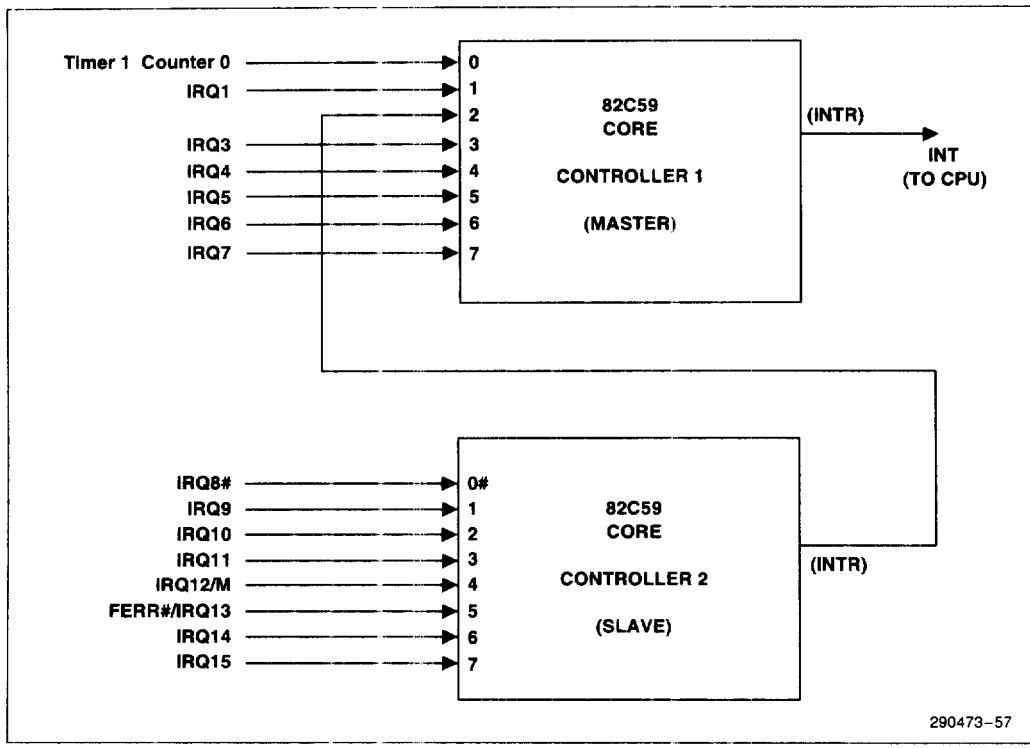


Figure 18. Block Diagram of the Interrupt Controller

Table 27 lists the I/O port address map for the interrupt registers:

Table 27. Interrupt Registers I/O Port Address Map

Interrupts	I/O Address	# of Bits	Register
IRQ[7:0]	0020h	8	CNTRL-1 Control Register
IRQ[7:0]	0021h	8	CNTRL-1 Mask Register
IRQ[15:8]	00A0h	8	CNTRL-2 Control Register
IRQ[15:8]	00A1h	8	CNTRL-2 Mask Register

IRQ0, IRQ2, (and possibly IRQ13 and IRQ12 if the "mouse" or floating point error logic is disabled in the ISA Clock Divisor Register), are connected to the interrupt controllers internally. The other interrupts are always generated internally and their typical functions are shown in Table 28:

Table 28. Typical Interrupt Functions

Priority	Label	Controller	Typical Interrupt Source
1	IRQ0	1	Interval timer 1, Counter 0 OUT
2	IRQ1	1	Keyboard
3-10	IRQ2	1	Interrupt from Controller 2
3	IRQ8#	2	Real Time Clock
4	IRQ9	2	Expansion Bus Pin B04
5	IRQ10	2	Expansion Bus Pin D03
6	IRQ11	2	Expansion Bus Pin D04
7	IRQ12/M	2	Mouse Interrupt
8	FERR#/IRQ13	2	Coprocessor Error
9	IRQ14	2	Fixed Disk Drive Controller Expansion Bus Pin D07
10	IRQ15	2	Expansion Bus Pin D06
11	IRQ3	1	Serial Port 2, Expansion Bus B25
12	IRQ4	1	Serial Port 1, Expansion Bus B24
13	IRQ5	1	Parallel Port 2, Expansion Bus B23
14	IRQ6	1	Diskette Controller, Expansion Bus B22
15	IRQ7	1	Parallel Port 1, Expansion Bus B21

5.8.1 EDGE AND LEVEL TRIGGERED MODES

There are two ELCR registers, one for each 82C59 bank. They are located at I/O ports 04D0h (for the Master Bank, IRQ[0:1,3:7]#) and 04D1h (for the Slave Bank, IRQ[8:15])#. They allow the edge and level sense selection to be made on an interrupt by interrupt basis instead of on a complete bank. Interrupts reserved for ISA use MUST be programmed for edge sensitivity (to ensure ISA compatibility). That is, IRQ (0,1,2,8#,13) must be programmed for edge sensitive operation. The LTIM bit (Edge/Level Bank select, offsets 20h, A0h) is disabled in the SIO. The default programming is equivalent to programming the LTIM bit (ICW1 bit 3) to a 0.

If an ELCR bit is equal to "0", an interrupt request will be recognized by a low to high transition on the corresponding IRQ input. The IRQ input can remain high without generating another interrupt.

If an ELCR bit is equal to "1", an interrupt request will be recognized by a "low" level on the corresponding IRQ input, and there is no need for an edge detection. For level triggered interrupt mode, the interrupt request signal must be removed before the EOI command is issued or the CPU interrupt must be disabled. This is necessary to prevent a second interrupt from occurring.

In both the edge and level triggered modes the IRQ inputs must remain active until after the falling edge of the first INTA#. If the IRQ input goes inactive before this time a DEFAULT IRQ7 will occur when the CPU acknowledges the interrupt. This can be a useful safeguard for detecting interrupts caused by spurious noise glitches on the IRQ inputs. To implement this feature the IRQ7 routine is used for "clean up" simply executing a return instruction, thus ignoring the interrupt. If IRQ7 is needed for other purposes a default IRQ7 can still be detected by reading the ISR. A normal IRQ7 interrupt will set the corresponding ISR bit, a default IRQ7 won't. If a default IRQ7 routine occurs during a normal IRQ7 routine, however, the ISR will remain set. In this case it is necessary to keep track of whether or not the IRQ7 routine was previously entered. If another IRQ7 occurs it is a default.

5.8.2 REGISTER FUNCTIONALITY

For a detailed description of the Interrupt Controller register set, please see Section 4.4, Interrupt Controller Register Description.

5.8.3 NON-MASKABLE INTERRUPT (NMI)

An NMI is an interrupt requiring immediate attention and has priority over the normal interrupt lines (IRQx). The SIO indicates error conditions by generating a non-maskable interrupt.

NMI interrupts are caused by the following conditions:

1. System Errors on the PCI Bus. SERR# is driven low by a PCI resource when this error occurs.
2. Parity errors on the add-in memory boards on the ISA expansion bus. IOCHK# is driven low when this error occurs.

The NMI logic incorporates two different 8-bit registers. These registers are addressed at locations 061h and 070h. The status of Port (061h) is read by the CPU to determine which source caused the NMI. Bits set to 1 in these ports show which device requested an NMI interrupt. After the NMI interrupt routine processes the interrupt, the NMI status bits are cleared by the software. This is done by setting the corresponding enable/disable bit high. Port (070H) is the mask register for the NMI interrupts. This register can mask the NMI signal and also disable or enable all NMI sources.

The individual enable/disable bits clear the NMI detect flip-flops when disabled.

All NMI sources can be enabled or disabled by setting Port 070h bit 7 to a 0 or 1. This disable function does not clear the NMI detect flip-flops. This means, if NMI is disabled then enabled via Port 070h, then an NMI will occur when Port 070h is re-enabled if one of the NMI detect flip-flops had been previously set.

To ensure that all NMI requests are serviced, the NMI service routine software needs to incorporate a few very specific requirements. These requirements are due to the edge detect circuitry of the host microprocessor, 80386 or 80486. The software flow would need to be the following:

1. NMI is detected by the processor on the rising edge of the NMI input.
2. The processor will read the status stored in port 061h to determine what sources caused the NMI. The processor may then set to 0 the register bits controlling the sources that it has determined to be active. Between the time the processor reads

- the NMI sources and sets them to a 0, an NMI may have been generated by another source. The level of NMI will then remain active. This new NMI source will not be recognized by the processor because there was no edge on NMI.

3. The processor must then disable all NMI's by setting bit 7 of port 070H to a 1 and then enable all NMI's by setting bit 7 of port 070H to a 0. This will cause the NMI output to transition low then high if there are any pending NMI sources. The CPU's NMI input logic will then register a new NMI.

Section 4.5 Control Registers, contains a detailed description of the NMI Status and Control Register (port 061h) and the NMI Enable and Real-Time Clock Address Register at port 070h.

5.9 Utility Bus Peripheral Support

The Utility Bus is a secondary bus buffered from the ISA Bus used to interface with peripheral devices that do not require a high speed interface. The buffer control for the lower 8 data signals is provided by the SIO via two control signals; UBUSOE# and UBUSTR. Figure 19 shows a block diagram of the external logic required as part of the decode and Utility Bus buffer control.

The SIO provides the address decode and three encoded chip selects to support:

1. Floppy Controller
2. Keyboard Controller
3. Real Time Clock
4. IDE Drive
5. 2 Serial Ports (COM1 and COM2)
6. 1 Parallel Port (LPT1, 2, or 3)
7. BIOS Memory
8. Configuration Memory (8 Kbyte I/O Mapped)

The SIO also supports the following functions:

1. Floppy DSKCHG Function
2. Port 92 Function (Alternate A20 and Alternate Reset)
3. Coprocessor Logic (FERR# and IGNNE# Function)

The binary code formed by the three Encoded Chip Selects determines which Utility Bus device is selected. The SIO also provides an Encoded Chip Select Enable signal (ECSEN#) that is used to select between the two external decoders. A zero selects decoder 1 and a one selects decoder 2. The table below shows the address decode for each of the Utility Bus devices.

Table 29. NMI Source Enable/Disable and Status Port Bits

NMI Source	I/O Port Bit for Status Reads	I/O Port Bit for Enable/Disable
IOCHK#	Port 061h, Bit 6	Port 061h, Bit 3
SERR#	Port 061h, Bit 7	Port 061h, Bit 2

Table 30. Encoded Chip Select Summary Table

ECSADDR2	ECSADDR1	ECSADDR0	ECSEN#	Address Decoded	External Chip Select	Note	Cycle Type
Decoder 1							
0	0	0	0	70h, 72h, 74, 76h	RTCALE#		I/O W
0	0	1	0	71h, 73h, 75h, 77h	RTCCS#		I/O R/W
0	1	0	0	60h, 62h, 64h, 66h	KEYBRDCS#		I/O R/W
0	1	1	0	000E0000h–000FFFFFh FFFE0000h–FFFFFFFFh FFF80000h–FFFDFFFFh	BIOSCS#	1	MEM R/W
1	0	0	0	3F0h–3F7h (primary) 370h–377h (secondary)	FLOPPYCS#	2	I/O R/W
1	0	1	0	1F0h–1F7h (primary) 170h–177h (secondary)	IDECS0#	2	I/O R/W
1	1	0	0	3F6h–3F7h (primary) 376h–377h (secondary)	IDECS1#	2	I/O R/W
1	1	1	0	Reserved			
Decoder 2							
0	0	0	1	Reserved			
0	0	1	1	0C00h	CPAGECS#	3	I/O R/W
0	1	0	1	0800h–08FFh	CFIGMEMCS#	3	I/O R/W
0	1	1	1	3F8h–3FFh (COM1) -or- 2F8h–37Fh (COM2)	COMACS#	4	I/O R/W
1	0	0	1	3F8h–3FFh (COM1) -or- 2F8h–37Fh (COM2)	COMBCS#	4	I/O R/W
1	0	1	1	3BCh–3BFh (LPT1) 378h–37Fh (LPT2) 278h–27Fh (LPT3)	LPTCS#	5	I/O R/W
1	1	0	1	Reserved			
1	1	1	1	Idle State			

NOTES:
1. The encoded chip select signals for BIOSCS# will always be generated for accesses to the upper 64 KB at the top of 1 MByte (F0000h–FFFFFh) and its aliases at the top of the 4 GB and 4 GB-1 MByte. Access to the lower 64 KByte (E0000h–EFFFFh) and its aliases at the top of 4 GB and 4GB–1MB can be enabled or disabled through the SIO. An additional 384 KB of BIOS memory at the top of 4 GB (FFFD0000h–FFFDFFFFh) can be enabled for BIOS use.
2. The primary and secondary locations are programmable through the SIO. Only one location range can be enabled at any one time. The floppy and IDE share the same enable and disable bit (i.e. if the floppy is set for primary, the IDE is also set for primary).
3. These signals can be used to select additional configuration RAM.
4. COM1 and COM2 address ranges can be programmed for either port A (COMACS#) or port B (COMBCS#).
5. Only one address range (LPT1, LPT2, or LPT3) can be programmed at any one time.

Port 92h Function

The SIO integrates the Port 92h Register. This register provides the alternate reset (ALTRST) and alternate A20 (ALT_A20) functions. Figure 19 shows how these functions are tied into the system.

DSKCHG Function

DSKCHG is tied directly to the DSKCHG signal of the floppy controller. This signal is inverted and driven onto system data line 7 (SD7) during I/O read cycles to floppy address locations 3F7h (primary) or 377 (secondary) as indicated by Table 31.

Table 31. DSKCHG Summary Table

FLOPPYCS# Decode	IDECSx# Decode	State of SD7 (Output)	State of UBUSOE#
Enabled	Enabled	Tri-stated	Enabled
Enabled	Disabled	Driven via DSKCHG	Disabled
Disabled	Enabled	Tri-stated	Enabled(1)
Disabled	Disabled	Tri-stated	Disabled

NOTE:
1. For this mode to be supported, extra logic is required to disable the U-bus transceiver for accesses to 3F7/377. This is necessary because of potential contention between the Utility bus buffer and a floppy on the ISA Bus driving the system bus at the same time during shared I/O accesses.

Coprocessor Error Support

If bit 5 in the ISA Clock Divisor Register is set to a one, the SIO will support coprocessor error reporting through the FERR#/IRQ13 signal.

FERR# is tied directly to the Coprocessor error signal of the CPU. If FERR# is driven active in this mode (coprocessor error detected by the CPU), an internal IRQ13 is generated and the INT output from the SIO is driven active. When a write to I/O location F0h is detected, the SIO negates IRQ13 and drives IGNNE# active. IGNNE# remains active until FERR# is driven inactive. Note that IGNNE# is not generated unless FERR# is active.

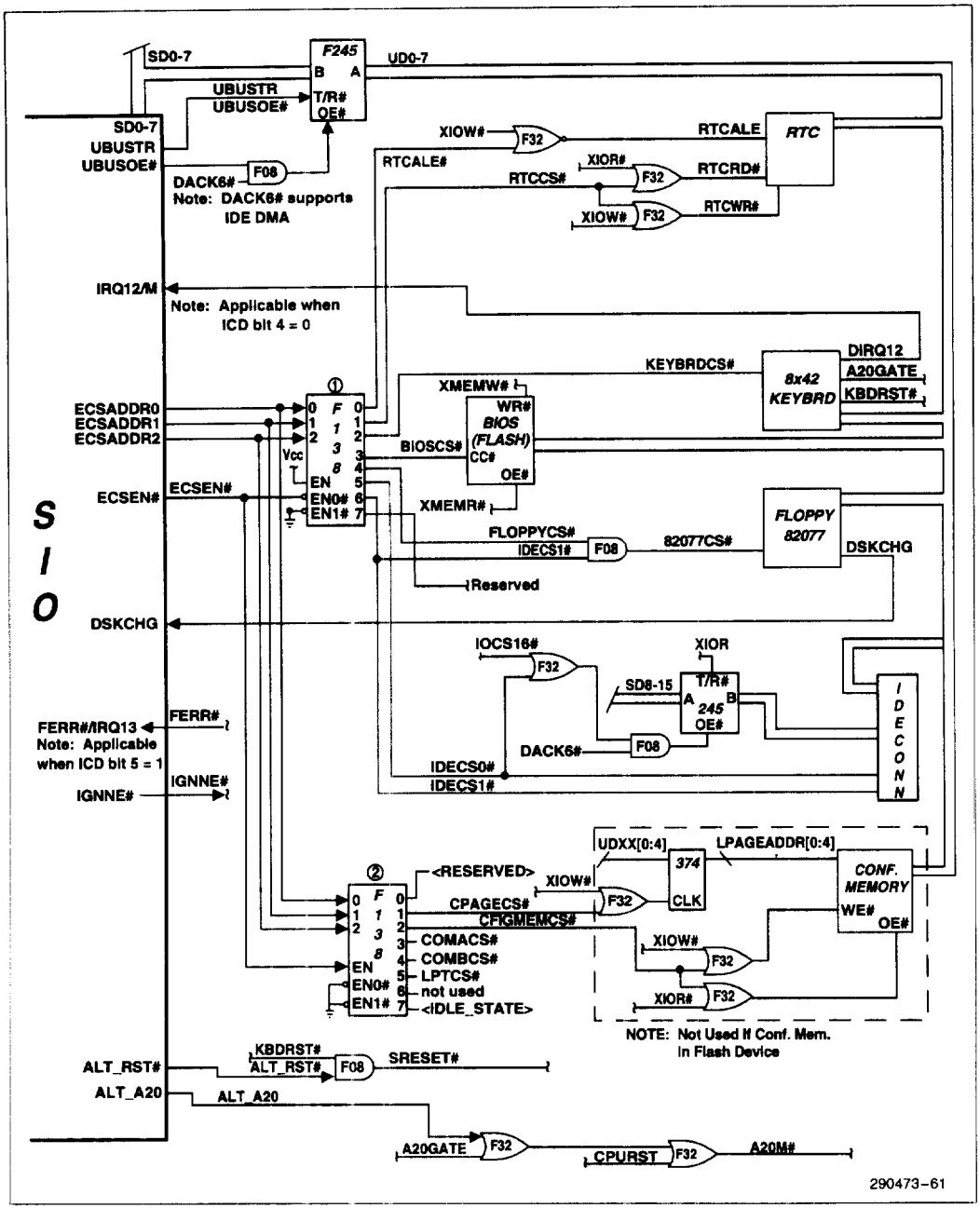


Figure 19. Utility Bus External Support Logic

Utility Bus accesses by the SIO, by an ISA master, and by the DMA is shown in Figure 20 and Figure 21. UBUSOE# and UBUSTR are driven differently for DMA cycles as shown in Figure 21.

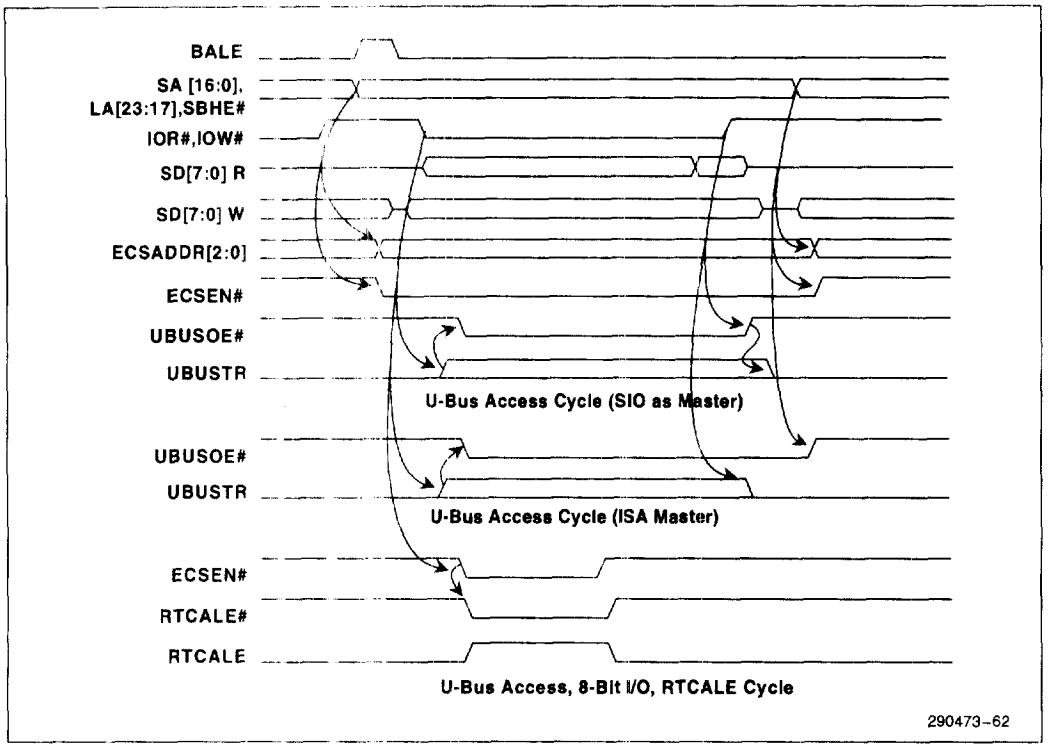


Figure 20. Utility Bus Access (SIO and ISA Master)

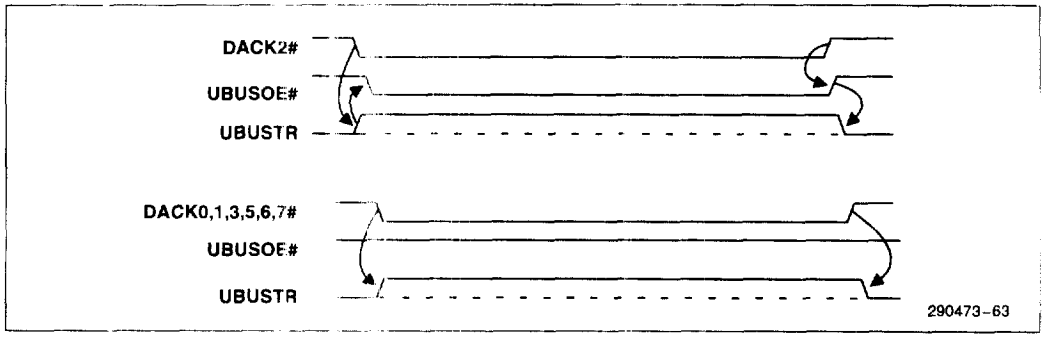


Figure 21. Utility Bus Access (DMA)

5.10 Power Management

The SIO's power management architecture is based around three core functions:

1. SMM (System Management Mode)
2. Clock Throttling
3. APM (Advanced Power Management Interface)

SMM is a mode during which an S Series Processor is executing SMM code from a secure memory space (SMRAM). SMM is invoked through the assertion of an SMI (System Management Interrupt). Physically, this is signaled over the SMI# pin. SMI's are triggered by various hardware and software events. SMRAM is used to store the SMM code which is really the SMI interrupt handler routine.

Clock Throttling will be used to reduce the power consumption of the CPU. STPCLK# is the physical signal used to control the CPU's clock.

APM creates an interface to allow the Operating System to communicate with the SMM code.

Figure 22 shows how the power management signals are connected in a Saturn based system with an S-series CPU.

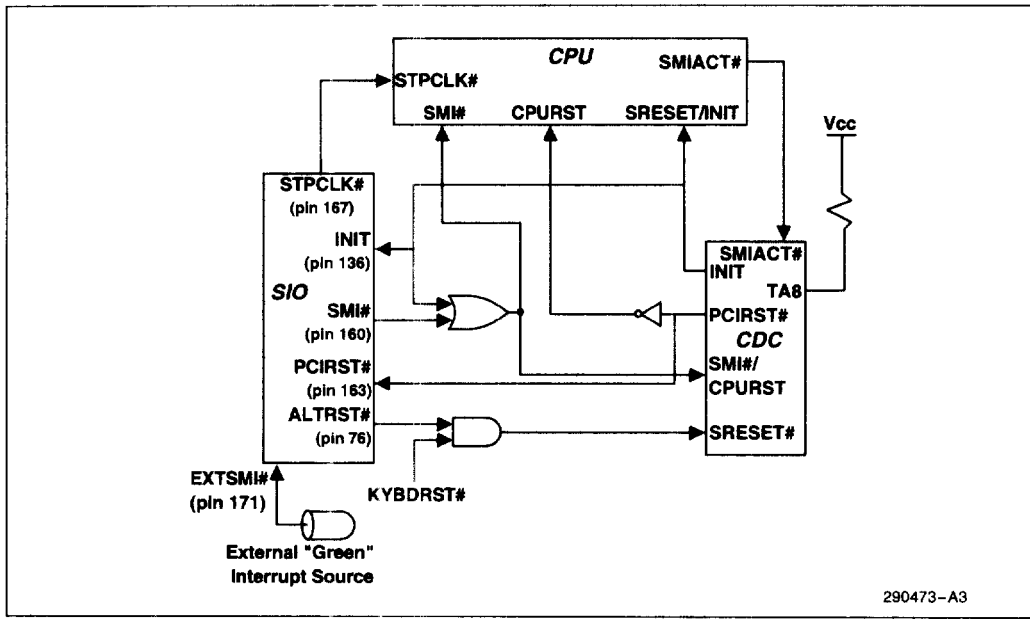


Figure 22. Power Management

6.0 ELECTRICAL CHARACTERISTICS

6.1 Absolute Maximum Ratings*

Case Temperature under Bias . . . −65°C to +110°C
Storage Temperature . . . . . −65°C to +150°C
Supply Voltages
with Respect to Ground . . . −0.5V to V_{CC} + 0.5V
Voltage On Any Pin . . . . . −0.5V to V_{CC} + 0.5V

** WARNING: Stressing the device beyond the "Absolute Maximum Ratings" may cause permanent damage. These are stress ratings only. Operation beyond the "Operating Conditions" is not recommended and extended exposure beyond the "Operating Conditions" may affect device reliability.*

7.0 MECHANICAL SPECIFICATIONS

7.1 Package Diagram

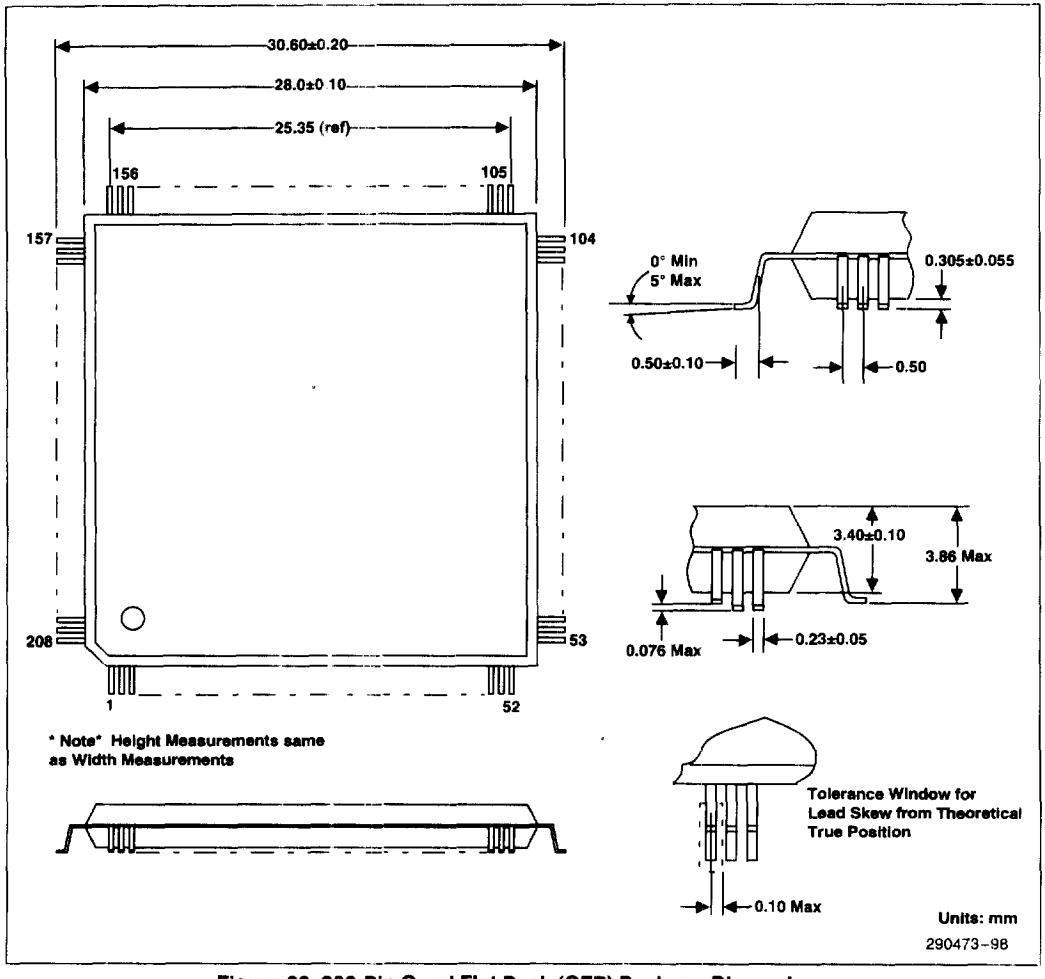


Figure 23. 208-Pin Quad Flat Pack (QFP) Package Dimensions

7.2 Thermal Specifications

Table 32. 82378 QFP Package Thermal Characteristics

Thermal Resistance-°C/Watt			
	Air Flow Rate (Ft./Min)		
Parameter	**0**	**200**	**400**
$\theta_{Junction\ to\ Case}$	6.6	6.6	6.6
$\theta_{Case\ to\ Ambient}$	36.6	27.4	24

8.0 TESTABILITY

The TEST and TESTO pins are used to test the SIO. During normal operations, the TEST pin must be grounded. The test output TESTO may be left as a no-connect (NC).

8.1 Global Tri-State

The TEST pin and IRQ3 are used to provide a high-impedance tri-state test mode. When the following input combination occurs, all outputs and bi-directional pins are tri-stated, with the exception of TESTO:

TEST = "1"
IRQ3 = "1"

The SIO must be reset after the bi-directional and output pins have been tri-stated in this manner.

8.2 NAND Tree

A NAND Tree is provided primarily for V_{IL}/V_{IH} testing. The NAND Tree is also useful for ATE at board level testing. The NAND Tree allows the tester to test the solder connections for each individual signal pin.

The TEST pin, along with IRQ5 or IRQ6, activates the NAND Tree. All bi-directional pins, and certain pure output pins using bi-directional buffers for performance reasons, are tri-stated when the following input combinations occur:

TEST = "1"
IRQ5 = "1"
- or -
TEST = "1"
IRQ6 = "0"

In the 82378, the output pulse train is observed at the TESTO test output. Pure output pins are not included directly in the NAND Tree. As noted in Section 8.3, each output can be expected to toggle after the corresponding node noted next to the pin name toggles from a "1" to a "0".

The sequence of the ATE test is as follows:

1. Drive TEST and IRQ5 high or TEST high and IRQ6 low.
2. Drive each input and bi-directional pin noted in Section 8.3 high.
3. Starting with the pin farthest from TESTO (SA8), individually drive each pin low. Expect TESTO to toggle with each pin. Expect each pure output noted in Section 8.3 to toggle after each corresponding input pin has been driven low.
4. Turn off tester drivers before driving TEST low.
5. Reset the SIO prior to proceeding with further testing.

8.3 NAND Tree Cell Order

Table 33. NAND Tree Cell Order

Tree Output #	Pin #	Pin Name	Notes
	14	IRQ4	Reserved
	21	TESTO	Test Mode Output
1	11	IRQ5	Cell Closest to TESTO
2	10	SA9	
3	9	IRQ6	
4	8	SA10	
5	7	IRQ7	
6	6	SA11	
7	5	SA12	
8	4	REFRESH#	
9	3	SA13	
10	207	SA14	
11	206	MASTER#	
12	205	SA15	
13	204	MEMW#	
14	203	MEMR#	
15	202	SA16	
16	201	SA17	
17	200	IOR#	
18	199	SA18	
19	198	IOW#	
20	197	SA19	
21	196	SMEMR#	
22	193	AEN	
23	192	SMEMW#	
24	191	IOCHRDY	
25	190	SD0	
26	189	SD1	

Table 33. NAND Tree Cell Order (Continued)

Tree Output #	Pin #	Pin Name	Notes
27	188	ZEROWS#	
28	187	SD2	
29	186	SD3	
30	185	SD4	
31	184	IRQ9	
32	180	SD5	
33	179	SD6	
34	178	SD7	
35	177	RSTDRV	
36	176	IOCHK#	
	175	ECSADDR0	NAND Tree Output of Tree Cell 28
	174	ECSADDR1	NAND Tree Output of Tree Cell 29
	173	ECSADDR2	NAND Tree Output of Tree Cell 30
37	172	IRQ8#	
38	171	EXTSMI#	
	170	ECSEN#	NAND Tree Output of Tree Cell 32
	169	TEST	PI = > VCC, TEST must be '1'
39	168	IRQ1	
	167	STPCLK#	
40	166	SYSCLK	
	165	UBUSTR	NAND Tree Output of Tree Cell 33
	164	UBUSOE#	NAND Tree Output of Tree Cell 34
41	163	PCIRST#	
42	161	DSKCHG	
	160	SMI#	
43	159	AD0	
44	155	AD1	
45	154	AD2	
46	153	AD3	

Table 33. NAND Tree Cell Order (Continued)

Tree Output #	Pin #	Pin Name	Notes
47	152	AD4	
48	151	AD5	
49	150	AD6	
50	149	AD7	
51	148	AD8	
52	147	C/BE0#	
53	146	AD9	
54	143	AD10	
55	142	AD11	
56	141	AD12	
57	140	AD13	
58	139	AD14	
59	138	AD15	
60	137	C/BE1#	
61	136	INIT	
62	135	PAR	
63	134	SERR#	
64	133	LOCK#	
65	132	STOP#	
66	128	DEVSEL#	
67	127	TRDY#	
68	126	IRDY#	
69	125	FRAME#	
70	124	C/BE2#	
71	123	AD16	
72	122	AD17	
73	121	AD18	
74	120	AD19	

Table 33. NAND Tree Cell Order (Continued)

Tree Output #	Pin #	Pin Name	Notes
75	119	AD20	
76	118	AD21	
77	115	AD22	
78	114	AD23	
79	113	C/BE3#	
80	112	AD24	
81	111	AD25	
82	110	AD26	
83	109	AD27	
84	108	AD28	
85	107	AD29	
86	106	AD30	
87	102	AD31	
88	101	IDSEL	
89	100	REQ3#	
90	98	REQ1#	
91	97	REQ2#	
92	96	CPUREQ#	
	95	CPUGNT#	NAND Tree Output of Tree Cell 93
	94	GNT1#	NAND Tree Output of Tree Cell 95
93	93	REQ0#	
	92	GNT0#	NAND Tree Output of Tree Cell 100
94	90	PCICLK	
	89	FLSHREQ#	NAND Tree Output of Tree Cell 102
95	88	MEMACK#	
	87	MEMREQ#	NAND Tree Output of Tree Cell 103

Table 33. NAND Tree Cell Order (Continued)

Tree Output #	Pin #	Pin Name	Notes
	86	MEMCS#	NAND Tree Output of Tree Cell 104
	85	ALT_A20	NAND Tree Output of Tree Cell 105
96	84	PIRQ[3]#	
97	83	PIRQ[2]#	
98	82	PIRQ[1]#	
99	81	PIRQ[0]#	
100	80	OSC	
	76	ALT_RST#	NAND Tree Output of Tree Cell 23
	75	INT	NAND Tree Output of Tree Cell 24
	74	NMI	NAND Tree Output of Tree Cell 25
101	73	SPKR	
	72	IGNNE#	NAND Tree Output of Tree Cell 26
102	71	FERR#	
103	70	SD15	
104	69	SD14	
105	68	SD13	
106	67	SD12	
107	65	DREQ7	
108	64	SD11	
109	63	DACK7#	
110	62	SD10	
111	61	DREQ6	
112	60	SD9	
113	59	DACK6#	
114	58	DREQ3	
115	57	DREQ2	
116	56	DREQ1	
117	55	SD8	
118	51	DREQ5	

Table 33. NAND Tree Cell Order (Continued)

Tree Output #	Pin #	Pin Name	Notes
119	50	DACK5#	
120	49	DACK3#	
121	48	DACK1#	
122	47	DREQ0	
123	46	LA17	
124	45	DACK0#	
125	44	LA18	
126	43	IRQ14	
127	42	LA19	
128	41	IRQ15	
129	40	LA20	
130	39	IRQ12/M	
131	38	LA21	
132	37	IRQ11	
133	36	LA22	
134	35	IRQ10	
135	34	LA23	
136	33	IOCS16#	
137	32	SBHE#	
138	31	MEMCS16#	
139	30	SA0	
140	29	SA1	
141	28	SA2	
142	24	SA3	
143	23	BALE	
144	22	SA4	
145	20	EOP	
146	19	SA5	

Table 33. NAND Tree Cell Order (Continued)

Tree Output #	Pin #	Pin Name	Notes
147	18	DACK2#	
148	17	SA6	
149	16	IRQ3	Output signals will transition from high-impedance state to driving state after this pin is driven low.
150	15	SA7	
151	13	SA8	Cell furthest from TESTO Start of NAND Tree

8.4 NAND Tree Diagram

Figure 24 shows the NAND Tree Diagram.

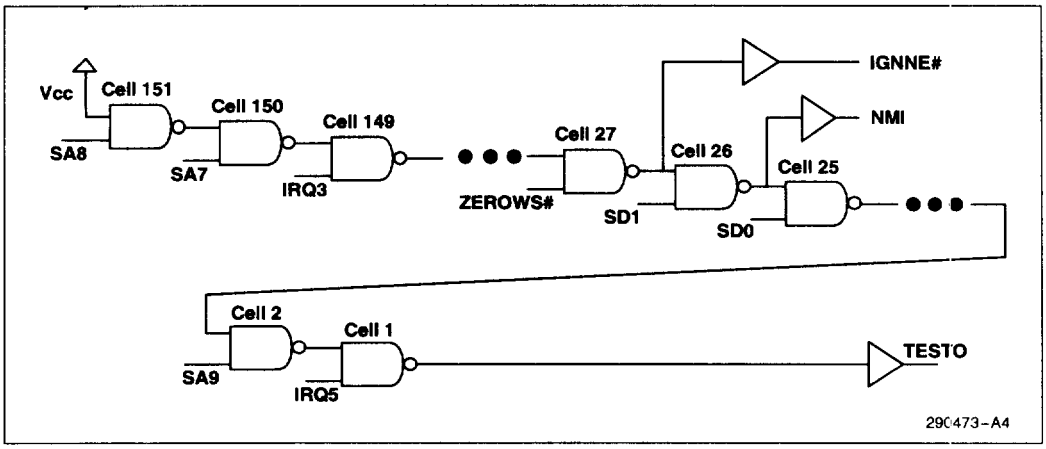


Figure 24. NAND Tree Diagram for 82378