INTRODUCTION TO THE PCI LOCAL BUS

The local bus concept was developed to break the PC data bottleneck. Traditional PC bus architectures are inadequate to meet the demands of today's graphics-oriented systems and large application sizes. A local bus moves peripherals off the I/O bus and places them closer to the system's processor bus, providing faster data transfer between the processor and peripherals.

The PCI Local Bus addresses the industry's need for a local bus standard that is not directly dependent on the speed and size of the processor bus, and that is both reliable and expandable. It represents the first time in the history of the PC industry that a common bus, independent of microprocessor design and manufacturer, has been adopted and used by rival PC computer architectures. PCI offers simple "plug and play" capability for the end user, and its performance is more than adequate for the most demanding applications, such as full-motion video.

FEATURES

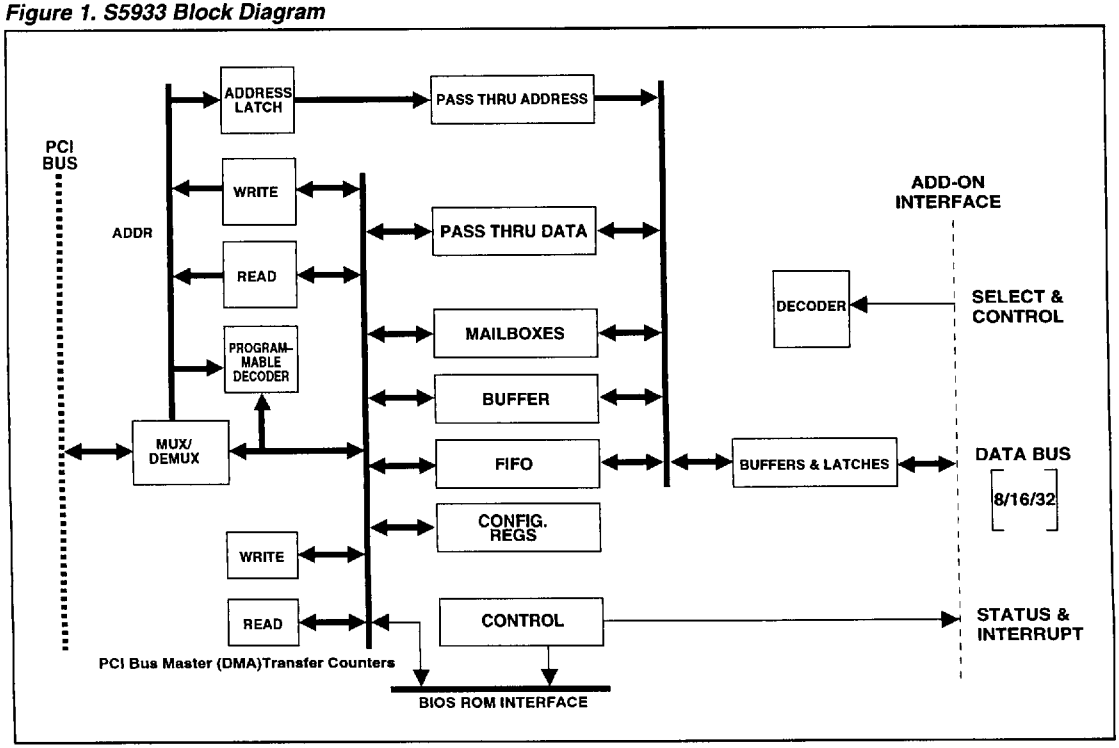
- PCI 2.1 Master/Slave Controller
- Generic 8/16/32-bit Add-On user bus
- Four definable memory block Pass-Thru regions
- Direct Add-On mailbox data strobe pin for PCI interrupt
- Optional boot load/external BIOS serial or byte-wide nvRAM
- Two 32 Byte FIFOs and 32 byte mailboxes
- Industry standard 160-pin PQFP

APPLICATIONS

- Digital Video
- Networking
- Multimedia
- Data Aquisition

3

Figure 1. S5933 Block Diagram



THE S5933 PCI CONTROLLER

AMCC's S5933 is a powerful and flexible PCI controller supporting several levels of interface sophistication. At the lowest level, it can serve simply as a PCI bus target with modest transfer requirements (since the PCI bus "plug-and-play" architecture is a benefit even when the transfer rate demand is low). For high-performance applications, the S5933 can become the bus master and can attain the peak transfer capabilities of 132 MByte/sec with a 32-bit PCI bus.

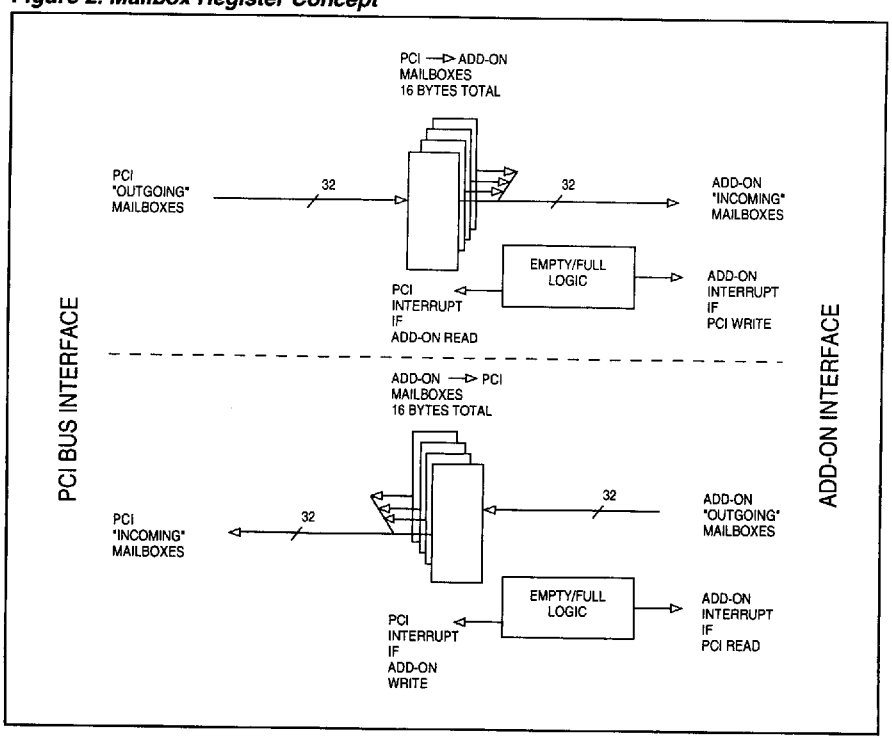
FUNCTIONAL ELEMENTS

The block diagram in Figure 1 shows the major functional elements within the S5933. The S5933 provides three physical bus interfaces: the PCI bus, the Add-On bus, and an optional external non-volatile memory. Data movement can occur between the PCI bus and the Add-On bus or the PCI bus and the non-volatile memory. Transfers between PCI and Add-On buses can take place through the mailbox registers or the FIFOs, or can make use of the Pass-Thru data path. FIFO transfers on the PCI bus interface can be performed either through software control or through hardware (using the S5933 as bus master).

MAILBOXES

The mailbox registers (shown in Figure 2) of the S5933 provide a bidirectional data path that can be used to send data or software control information between the system platform and the Add-On product. These mailboxes are intended to serve as customizable command, status, and parametric data registers. Use of the mailbox registers is defined by an application's own software; they are often used to initiate larger data transfers over either the FIFO or Pass-Thru data paths. Interrupts can be configured to occur on the PCI and Add-On bus (if desired) based on a specific mailbox event(s).

Figure 2. Mailbox Register Concept



FIFOs

Two separate FIFO data paths exist within the S5933. One FIFO handles data movement from the PCI bus to the Add-On bus (shown in Figure 3) and the other handles movement in the opposite direction (Figure 4). Both of the FIFOs are able to support PCI bus mastering; each has an address pointer and transfer count register associated with its PCI bus transaction. An important feature of the S5933 is the ability to optionally perform various endian translations when data is moved through the FIFO. This feature permits a single Add-On product to maintain a fixed endian structure within the Add-On while allowing the system platform to operate in its own native endian format.

Figure 3. PCI to Add-On FIFO Concept

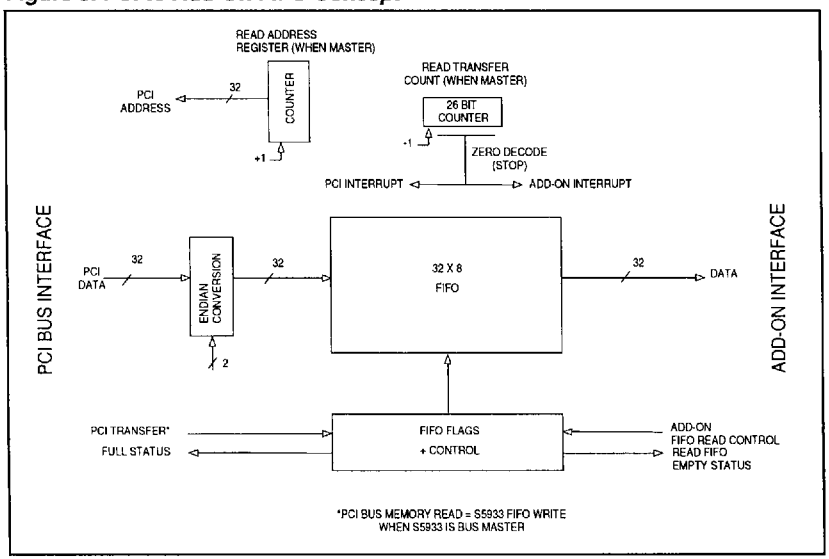
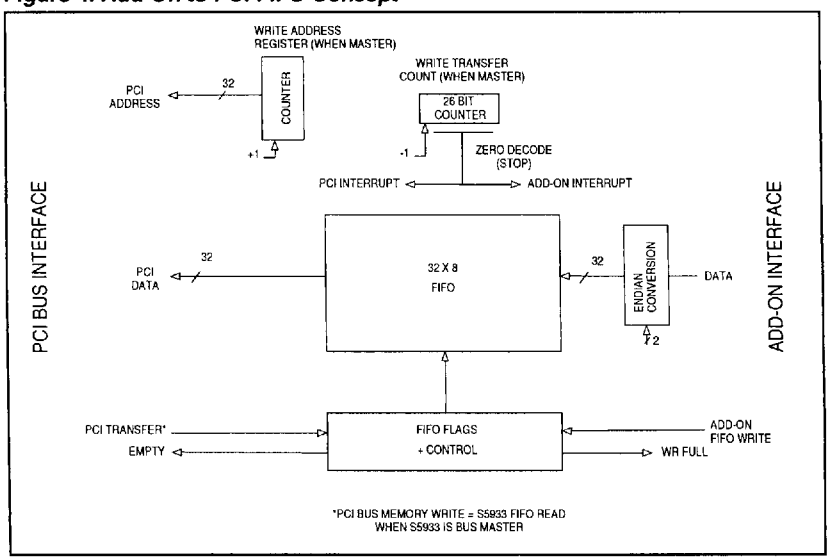


Figure 4. Add-On to PCI FIFO Concept



PASS-THRU

Figure 5 depicts the Pass-Thru concept that allows actual PCI bus transactions to be executed in real time with the Add-On interface. The Pass-Thru feature can be used to achieve high-performance burst transfers between the PCI bus and Add-On bus external peripherals or memory devices. In the Pass-Thru mode, the S5933 provides the benefits of the configuration registers and full PCI Bus Specification 2.1 compliance, while permitting customization and flexibility in the implementation of the Add-On product.

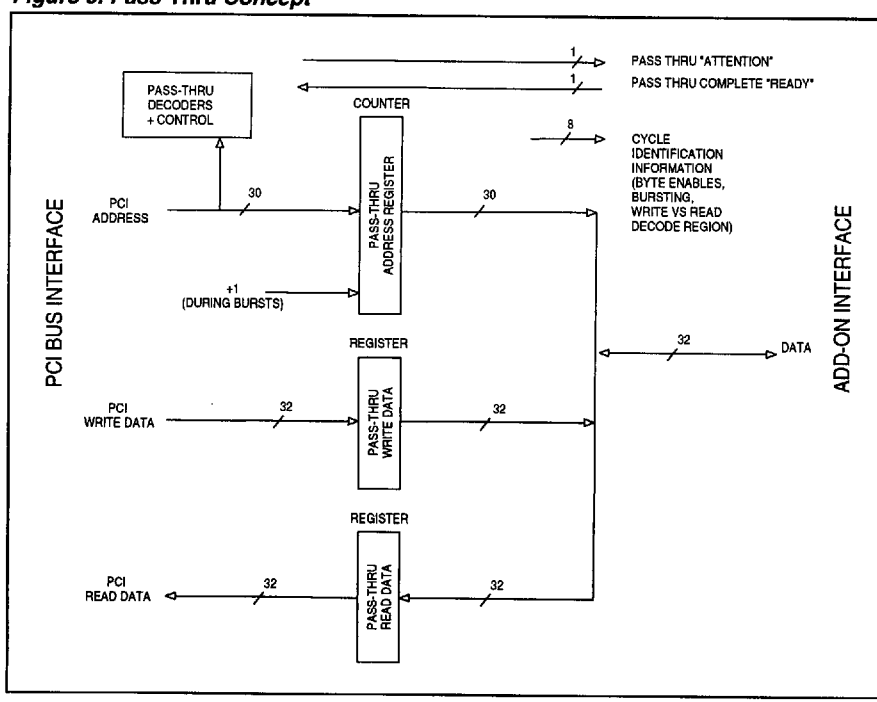
The Pass-Thru logic is comprised of one address register and two data registers (one for each direction). Control information necessary to define the current (Pass-Thru) PCI bus transaction is also provided to the Add-On interface on dedicated S5933 package pins.

PCI AGENT

The S5933 was designed to serve as an "endpoint" within a given PCI system. This means that it is the final point in a data transfer (for example, a video screen, network, sound output, or storage device) and/or the beginning point of a data transfer (for example, video source, network, audio source, or storage device). Some PCI elements, termed "bridges," allow for data transactions to span to other bus standards and therefore are not "endpoints."

As a PCI bus agent, the S5933 is required to support the configuration registers in order to be compliant with the PCI Specification. These configuration registers provide for a standardized method for system platform software to integrate Add-On functions in a machine. Another important aspect of the standard is that all PCI bus agents can be controlled (enabled/disabled, assigned physical address space, etc.) in a common manner, regardless of function, by the hosting system. These configuration registers defined by the PCI standard (and present within the S5933) are described in the PCI Configuration Chapter.

Figure 5. Pass-Thru Concept



ADD-ON INTERFACE

The S5933 provides a simple, general-purpose interface to the Add-On bus. The Add-On data path is a 32-bit bus for the S5933. Data transfers to/from the S5933 internal registers are accomplished through a chip select decode in conjunction with either a read or write strobe. The S5933 provides dedicated pins which allow its FIFOs to be used in the implementation of custom DMA ports or additional external FIFOs (if necessary).

The output pins on the Add-On interface include an interrupt source, a buffered clock, and a software-controllable reset. The interrupt output pin is provided to signal when a selected mailbox or self-test event occurs from the PCI interface. The buffered clock output is a possible Add-On cost reduction feature, and in addition provides synchronization for Pass-Thru data transfers. The software-controllable reset from the S5933 provides Add-On hardware with a means for proper handling of a system's "soft" reboot (e.g.CTRL-ALT-DEL).

NON-VOLATILE MEMORY INTERFACE

The non-volatile interface allows customization of the S5933 and provides for an optional BIOS ROM on the PCI bus. The non-volatile memory can be either a serial device or a byte-wide device. Serial devices may range from 128 bytes through 2048 bytes, and byte-wide devices from 128 bytes through 65,536 bytes.

As an example, the Vendor and Device ID numbers in the PCI configuration space can be initialized to reflect values contained in the external non-volatile memory. After initialization, a user's own Vendor and Device ID is then presented by the S5933 when requested by the PCI bus.

For some non-volatile devices, it is possible to write the non-volatile memory from the PCI interface. This feature lets the system designer establish a field upgrade strategy for the BIOS software or a method to support various system platforms with only one Add-On board product.

Use of a byte-wide non-volatile memory device is shown in Figure 6; Figure 7 shows a serial device with the S5933.

3

Figure 6. Byte-Wide Non-Volatile Memory Interface

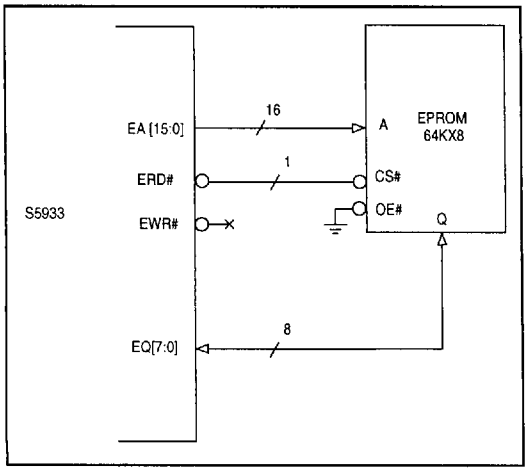


Figure 7. Serial Non-Volatile Memory Interface

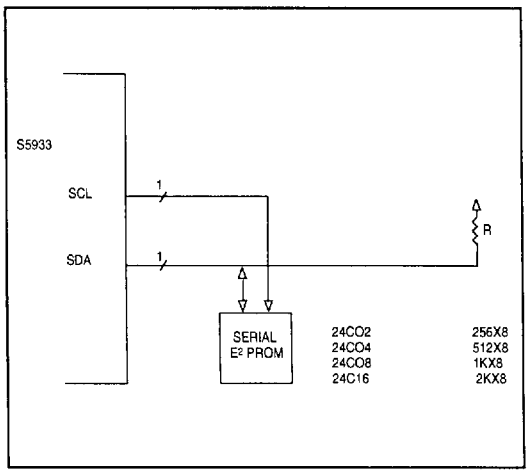
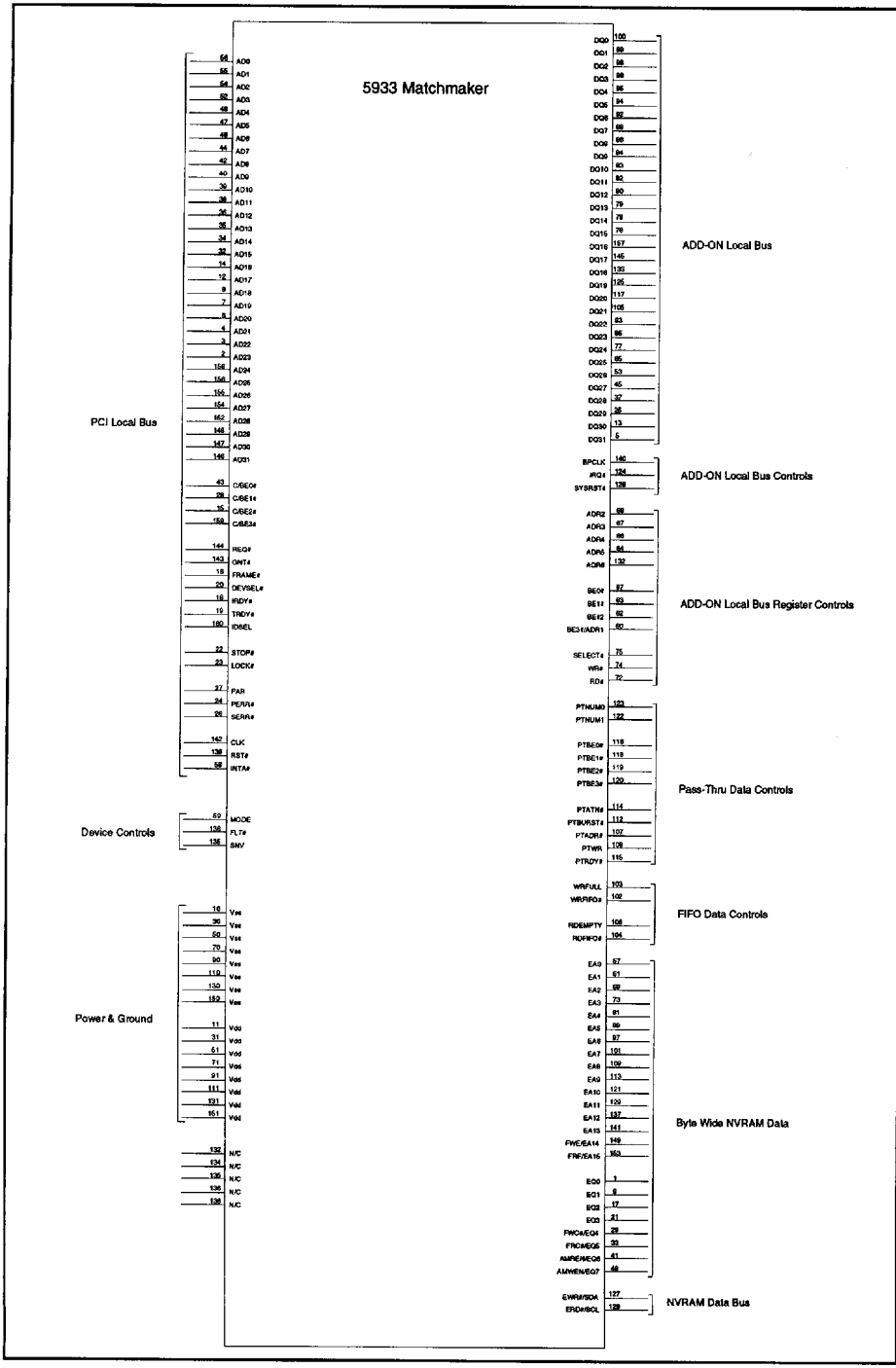


Figure 8. S5933 Pin Assignment



Signal Type Definition

The following signal type definitions [in, out, t/s, s/t/s and o/d] are taken from Revision 2.1 of the PCI local bus specification.

in Input is a standard input-only signal.

out Totem Pole Output is a standard active driver.

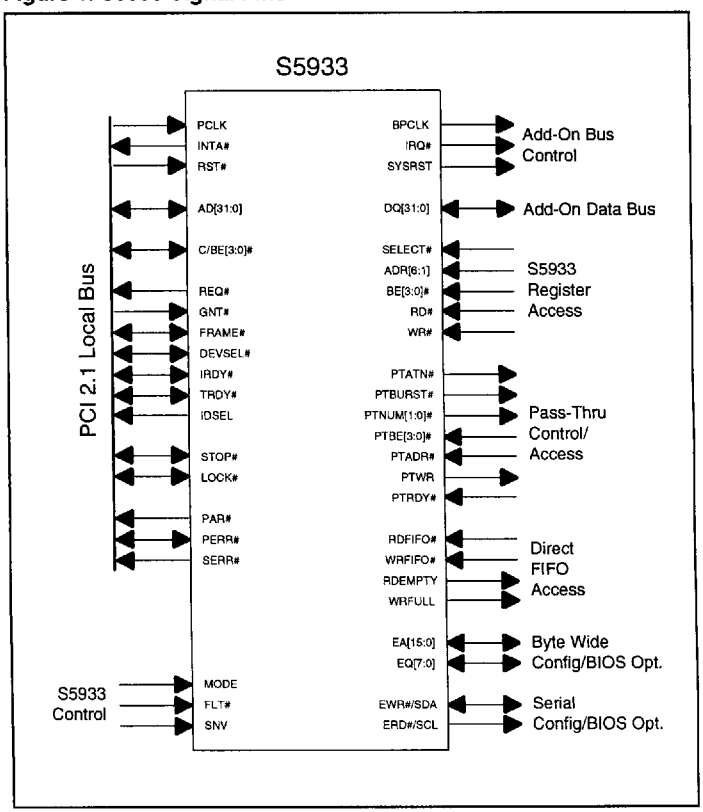
t/s Tri-State® is a bidirectional, tristate input/output pin.

s/t/s Sustained Tri-State is an active low tristate signal owned and driven by one and only one agent at a time. The agent that drives an s/t/s pin low must drive it high for at least one clock before letting it float. A new agent cannot start driving a s/t/s signal any sooner than one clock after the previous owner tri-states it. A pullup is required to sustain the inactive state until another agent drives it, and must be provided by the central source.

o/d Open Drain allows multiple devices to share as a wire-OR.

Note that a # symbol at the end of a signal name denotes that the active state occurs when the signal is at a low voltage. When no # symbol is present, the signal is active high.

Figure 1. S5933 Signal Pins



3

PCI BUS INTERFACE SIGNALS

Address and Data Pins — PCI Local Bus

<table>
<tr><th>Signal</th><th>Type</th><th>Description</th></tr>
<tr><td>AD[31:00]</td><td>t/s</td><td>Local Bus Address/Data lines. Address and data are multiplexed on the same pins. Each bus operation consists of an address phase followed by one or more data phases. Address phases are identified when the control signal, FRAME#, is asserted. Data transfers occur during those clock cycles in which control signals IRDY# and TRDY# are both asserted.</td></tr>
<tr><td>C/BE[3:0]#</td><td>t/s</td><td>Bus Command and Byte Enables. These are multiplexed on the same pins. During the address phase of a bus operation, these pins identify the bus command, as shown in the table below. During the data phase of a bus operation, these pins are used as Byte Enables, with C/BE[0]# enabling byte 0 (least significant byte) and C/BE[3]# enabling byte 3 (most significant byte).
<table>
<tr><th colspan="4">C/BE[3:0]#</th><th>Description
(during address phase)</th></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>Interrupt Acknowledge</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>1</td><td>Special Cycle</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>I/O READ</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>1</td><td>I/O WRITE</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>0</td><td>Reserved</td></tr>
<tr><td>0</td><td>1</td><td>0</td><td>1</td><td>Reserved</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>0</td><td>Memory Read</td></tr>
<tr><td>0</td><td>1</td><td>1</td><td>1</td><td>Memory Write</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>0</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>1</td><td>Reserved</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>0</td><td>Configuration Read</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>1</td><td>Configuration Write</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>0</td><td>MEMORY READ - Multiple</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>1</td><td>Dual Address Cycle</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>0</td><td>Memory Read Line</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>1</td><td>Memory Write and Invalidate</td></tr>
</table></td></tr>
<tr><td>PAR</td><td>t/s</td><td>Parity. This signal is even parity across the entire AD[31:00] field along with the C/BE[3:0]# field. The parity is stable in the clock following the address phase and is sourced by the master. During the data phase for write operations, the bus master sources this signal on the clock following IRDY# active; during the data phase for read operations, this signal is sourced by the target and is valid on the clock following TRDY# active. The PAR signal therefore has the same timing as AD[31:00}, delayed by one clock.</td></tr>
</table>

System Pins — PCI Local Bus

Signal	Type	Description
CLK	in	Clock. The rising edge of this signal is the reference upon which all other signals are based, with the exception of RST# and the interrupt (IRQA#-). The maximum frequency for this signal is 33 MHz and the minimum is DC (0 Hz).
RST#	in	Reset. This signal is used to bring all other signals within this device to a known, consistent state. All PCI bus interface output signals are not driven (tri-stated), and open drain signals such as SERR# are floated.

Interface Control Pins — PCI Bus Signal

Signal	Type	Description
FRAME#	s/t/s	Frame. This signal is driven by the current bus master and identifies both the beginning and duration of a bus operation. When FRAME# is first asserted, it indicates that a bus transaction is beginning and that valid addresses and a corresponding bus command are present on the AD[31:00] and C/BE[3:0] lines. FRAME# remains asserted during the data transfer portion of a bus operation and is deasserted to signify the final data phase.
IRDY#	s/t/s	Initiator Ready. This signal is sourced by the bus master and indicates that the bus master is able to complete the current data phase of a bus transaction. For write operations, it indicates that valid data is on the AD[31:00] pins. Wait states occur until both TRDY# and IRDY# are asserted together.
TRDY#	s/t/s	Target Ready. This signal is sourced by the selected target and indicates that the target is able to complete the current data phase of a bus transaction. For read operations, it indicates that the target is providing valid data on the AD[31:00] pins. Wait states occur until both TRDY# and IRDY# are asserted together.
STOP#	s/t/s	Stop. The Stop signal is sourced by the selected target and conveys a request to the bus master to stop the current transaction.
LOCK#	in	Lock. The lock signal provides for the exclusive use of a resource. The S5933 may be locked as a target by one master at a time. The S5933 cannot lock a target when it is a master.
IDSEL	in	Initialization Device Select. This pin is used as a chip select during configuration read or write operations.
DEVSEL#	s/t/s	Device Select. This signal is sourced by an active target upon decoding that its address and bus commands are valid. For bus masters, it indicates whether any device has decoded the current bus cycle.

Arbitration Pins (Bus Masters Only) — PCI Local Bus

Signal	Type	Description
REQ#	out	Request. This signal is sourced by an agent wishing to become the bus master. It is a point-to-point signal and each master has its own REQ#.
GNT#	in	Grant. The GNT# signal is a dedicated, point-to-point signal provided to each potential bus master and signifies that access to the bus has been granted.

Error Reporting Pins — PCI Local Bus

Signal	Type	Description
PERR#	s/t/s	Parity Error. This pin is used for reporting parity errors during the data portion of a bus transaction for all cycles except a Special Cycle. It is sourced by the agent receiving data and driven active two clocks following the detection of the error. This signal is driven inactive (high) for one clock cycle prior to returning to the tri-state condition.
SERR#	o/d	System Error. This pin is used for reporting address parity errors, data parity errors on Special Cycle commands, or any error condition having a catastrophic system impact.

Interrupt Pin — PCI Local Bus

Signal	Type	Description
INTA#	o/d	Interrupt A. This pin is a level sensitive, low active interrupt to the host. The INTA# interrupt must be used for any single function device requiring an interrupt capability.

NON-VOLATILE MEMORY INTERFACE SIGNALS

This signal grouping provides for connection to external non-volatile memories. Either a serial or byte-wide device may be used.

The serial interface shares the read and write control pins used for interfacing with byte-wide memory devices. Since it is intended that only one (serial or byte wide) configuration be used in any given implementation, separate descriptions are provided for each. The S5933 provides the pins necessary to interface to a byte wide non-volatile memory. When they are connected to a properly configured serial memory, these byte wide interface pins assume alternate functions. These alternate functions include added external FIFO status flags, FIFO reset control, Add-On control for bus mastering and a hardware interface mailbox port.

Serial nv Devices

Signal	Type	Description
SCL	t/s	Serial Clock. This output is intended to drive a two-wire Serial Interface and functions as the bus's master. It is intended that this signal be directly connected to one or more inexpensive serial non-volatile RAMs or EEPROMs. This pin is shared with the byte wide interface signal, ERD#.
SDA	t/s	Serial Data/Address. This bidirectional pin is used to transfer addresses and data to or from a serial nvRAM or EEPROM. It is an open drain output and intended to be wire-ORed with all other devices on the serial bus using a 4.7K external pull-up resistor. This pin is shared with the byte wide interface signal, EWR#.
SNV	in	Serial Non-Volatile Device. This input, when high, indicates a serial boot device or no boot device is present. When this pin is low, a byte-wide boot device is present. Note: SCL and SDA are not controlled by FLT#.

Byte-Wide nv Devices

Signal	Type	Description
EA[15:00]	t/s	External nv memory address. These signals connect directly to the external BIOS (or EEPROM) or EPROM address pins EA0 through EA15. The PCI interface controller assembles 32-bit-wide accesses through multiple read cycles of the 8-bit device. The address space from 0040h through 007Fh is used to preload and initialize the PCI configuration registers. Should an external nv memory be used, the minimum size required is 128 bytes and the maximum is 64K bytes. When a serial memory is connected to the S5933, the pins EA[7:0] are reconfigured to become a hardware Add-On to PCI mailbox register with the EA8 pin as the mailbox load clock. Also, the EA15 signal pin will provide an indication that the PCI to Add-On FIFO is full (FRF), and the EA14 signal pin will indicate whether the Add-On to PCI FIFO is empty (FWE).
ERD#	out	External nv memory read control. This pin is asserted during read operations involving the external non-volatile memory. Data is transferred into the S5933 during the low to high transition of ERD#. This pin is shared with the serial external memory interface signal, SCL.
EWR#	t/s	External nv memory write control. This pin is asserted during write operations involving the external non-volatile memory. Data is presented on pins EQ[7:0] along with its address on pins EA[15:0] throughout the entire assertion of EWR#. This pin is shared with the serial external memory interface signal, SDA.
EQ[7:0]	t/s	External memory data bus. These pins are used to directly connect with the data pins of an external non-volatile memory. When a serial memory is connected to the S5933, the pins EQ4, EQ5, EQ6 and EQ7 become reconfigured to provide signal pins for bus mastering control from the Add-On interface.

ADD-ON BUS INTERFACE SIGNALS

The following sets of signals represent the interface pins available for the Add-On function. There are four groups: Register access, FIFO access, Pass-Thru mode pins, and general system pins.

Register Access Pins

Signal	Type	Description
DQ[31:00]	t/s	Datapath DQ0–DQ31. These pins represent the datapath for the Add-On peripheral's data bus. They provide the interface to the controller's FIFO and other registers. When MODE=V_{CC}, only DQ[15:00] are used. DQ[31:0] have internal pull-up resistors.
ADR[6:2]	in	Add-On Addresses. These signals are the address lines to select which of the 16 DWORD registers within the controller is desired for a given read or write cycle, as shown in the table below. ADR6 has an internal pull-down resistor.
BE3# or ADR1	in	Byte Enable 3 (32-bit mode) or ADR1 (16 bit mode). This pin is used in conjunction with the read or write strobes (RD# or WR#) and the Add-On select signal, SELECT#. As a Byte Enable, it is necessary to have this pin asserted to perform write operations to the register identified by ADR[6:2] bit locations d24 through d31; for read operations it controls the DQ[31:24] output drive. BE3# has an internal pull-up resistor.
BE[2:0]#	in	Byte Enable 2 through 0. These pins provide for individual byte control during register read or write operations. BE2# controls activity over DQ[23:DQ16], BE1# controls DQ[15:8], and BE0# controls DQ[7:0]. During read operations they control the output drive for each of their respective byte lanes; for write operations they serve as a required enable to perform the modification of each byte lane. BE[2:0]# have internal pull-up resistors.
SELECT#	in	Select for the Add-On interface. This signal must be driven low for any write or read access to the Add-On interface registers. This signal must be stable during the assertion of command signals WR# or RD#.
WR#	in	Write strobe. This pin, when asserted in conjunction with the SELECT# pin, causes the writing of one of the internal registers. The specific register and operand size are identified through address pins ADR[6:2] and the byte enables, BE[3:0]#.
RD#	in	Read strobe. This pin, when asserted in conjunction with the SELECT# pin, causes the reading of one of the internal registers. The specific register and operand size are identified through address pins ADR[6:2] and the byte enables BE[3:0]#.
MODE	in	This pin control whether the S5933 data accesses on the DQ bus are to be 32-bits wide (MODE = low) or 16-bits wide (MODE = high). When in the 16 bit mode, the signal BE3# is reassigned as the address signal ADR1. MODE has an internal pull-down resistor.

ADR[6:2]	Register Name
0 0 0 0 0	Add-On Incoming Mailbox Reg. 1
0 0 0 0 1	Add-On Incoming Mailbox Reg. 2
0 0 0 1 0	Add-On Incoming Mailbox Reg. 3
0 0 0 1 1	Add-On Incoming Mailbox Reg. 4
0 0 1 0 0	Add-On Outgoing Mailbox Reg. 1
0 0 1 0 1	Add-On Outgoing Mailbox Reg. 2
0 0 1 1 0	Add-On Outgoing Mailbox Reg. 3
0 0 1 1 1	Add-On Outgoing Mailbox Reg. 4
0 1 0 0 0	Add-On FIFO Port
0 1 0 0 1	Bus Master Write Address Register
0 1 0 1 0	Add-On Pass-Thru Address
0 1 0 1 1	Add-On Pass-Thru Data
0 1 1 0 0	Bus Master Read Address Register
0 1 1 0 1	Add-On Mailbox Empty/Full Status
0 1 1 1 0	Add-On Interrupt Control
0 1 1 1 1	Add-On General Control/Status Register
1 0 1 1 0	Bus Master Write Transfer Count
1 0 1 1 1	Bus Master Read Transfer Count

FIFO Access Pins

Signal	Type	Description
WRFIFO#	in	Write FIFO. This signal provides a method to directly write the FIFO without having to generate the SELECT# signal or the ADR[6:2] value of [01000b] to access the FIFO. Access width is either 32 bits or 16 bits depending on the data bus size available. This signal is intended for implementing PCI DMA transfers with the Add-On system. This pin has an internal pull-up resistor.
RDFIFO#	in	Read FIFO. This signal provides a method to directly read the FIFO without having to generate the SELECT# signal or the ADR[6:2] value of [01000b] to access the FIFO. Access width is either 32 bits or 16 bits, depending on the data bus size defined by the MODE pin. This signal is intended for implementing PCI DMA transfers with the Add-On system. This pin has an internal pull-up resistor.
WRFULL	out	Write FIFO full. This pin indicates whether the Add-On-to-PCI bus FIFO is able to accept more data. This pin is intended to be used to implement DMA hardware on the Add-On system bus. A logic low output from this pin can be used to represent a DMA write (Add-On to-PCI FIFO) request.
RDEMPTY	out	Read FIFO Empty. This pin indicates whether the read FIFO (PCI-to-Add-On FIFO) contains data. This pin is intended to be used by the Add-On system to control DMA transfers from the PCI bus to the Add-On system bus. A logic low from this pin can be used to represent a DMA (PCI-to-Add-On FIFO) request.

3

Pass-Thru Interface Pins

Signal	Type	Description
PTATN#	out	Pass-Thru Attention. This signal identifies that an active PCI bus cycle has been decoded and data must be read from or written to the Pass-Thru Data Register.
PTBURST#	out	Pass-Thru Burst. This signal identifies PCI bus operations involving the current Pass-Thru cycle as requesting burst access.
PTRDY#	in	Pass-Thru Ready. This input indicates when Add-On logic has completed a Pass-Thru cycle and another may be initiated.
PTNUM[1:0]	out	Pass-Thru Number. These signals identify which of the four base address registers decoded a Pass-Thru bus activity. These bits are only meaningful when signal PTATN# is active. A value of 00 corresponds to Base Address Register 1, a value of 01 for Base Address Register 2, and so on.
PTBE[3:0]#	out	Pass-Thru Byte Enables. These signals indicate which bytes are requested for a given Pass-Thru operation. They are valid during the presence of signal PTATN# active.
PTADR#	in	Pass-Thru Address. This signal causes the actual Pass-Thru requested address to be presented as outputs on the DQ pins DQ[31:0] for Add-Ons with 32-bit buses, or the low-order 16 bits for Add-Ons with 16-bit buses. It is necessary that all other bus control signals be in their inactive state during the assertion of PTADR#. The purpose of this signal is to provide the direct addressing of external Add-On peripherals through use of the PTNUM[1:0] and the low-order address bits presented on the DQ bus with this pin active.
PTWR	out	Pass-Thru Write. This signal identifies whether a Pass-Thru operation is a read or write cycle. This signal is valid only when PTATN# is active.

System Pins

Signal	Type	Description
SYSRST#	out	System Reset. This low active output is a buffered form of the PCI bus reset, RST#. It is not synchronized to any clock within the PCI interface controller. Additionally, this signal can be invoked through software from the PCI host interface.
BPCLK	out	Buffered PCI Clock. This output is a buffered form of the PCI bus clock and, as such, has all of the behavioral characteristics of the PCI clock (i.e., DC-to-33 MHz capability).
IRQ#	out	Interrupt. This pin is used to signal the Add-On system that a significant event has occurred as a result of activity within the PCI controller.
FLT#	in	Float. When asserted, all S5933 outputs are floated. This pin has an internal pull-up resistor.

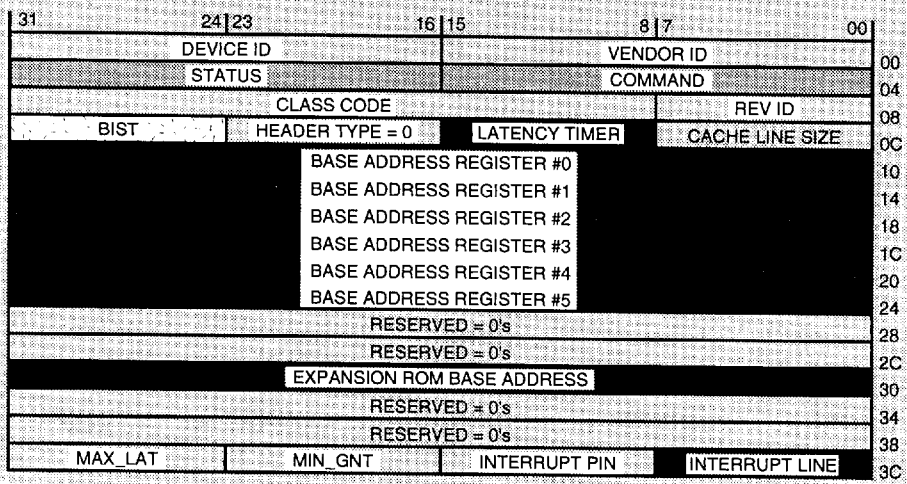
PCI CONFIGURATION REGISTERS

Each PCI bus device contains a unique 256-byte region called its configuration header space. Portions of this configuration header are mandatory in order for a PCI agent to be in full compliance with the PCI specification. This section describes each of the configuration space fields—its address, default values, initialization options, and bit definitions—and also provides an explanation of its intended usage.

Table 1. Configuration Registers

Configuration Address Offset	Abbreviation	Register Name
00h–01h	VID	Vendor Identification
02h–03h	DID	Device Identification
04h–05h	PCICMD	PCI Command Register
06h–07h	PCISTS	PCI Status Register
08h	RID	Revision Identification Register
09h–0Bh	CLCD	Class Code Register
0Ch	CALN	Cache Line Size Register
0Dh	LAT	Master Latency Timer
0Eh	HDR	Header Type
0Fh	BIST	Built-in Self-test
10h–27h	BADR0-BADR5	Base Address Registers (0-5)
28h–2Fh	—	Reserved
30h	EXROM	Expansion ROM Base Address
34h–3Bh	—	Reserved
3Ch	INTLN	Interrupt Line
3Dh	INTPIN	Interrupt Pin
3Eh	MINGNT	Minimum Grant
3Fh	MAXLAT	Maximum Latency
40h–FFh	—	Not used

PCI Configuration Space Header



LEGEND

- EPROM IS DATA SOURCE (READ ONLY)
- CONTROL FUNCTION
- EPROM INITIALIZED RAM (CAN BE ALTERED FROM PCI PORT)
- EPROM INITIALIZED RAM (CAN BE ALTERED FROM ADD-ON PORT)
- HARD-WIRED TO ZEROES

Note: Some registers are a combination of the above. See individual sections for full description.

VENDOR IDENTIFICATION REGISTER (VID)

Register Name: Vendor Identification
Address Offset: 00h-01h
Power-up value: 10E8h (AMCC, Applied Micro Circuits Corp.)
Boot-load: External nvRAM offset 040h-41h
Attribute: Read Only (RO)
Size: 16 bits

The VID register contains the vendor identification number. This number is assigned by the PCI Special Interest Group and is intended to uniquely identify any PCI device. Write operations from the PCI interface have no effect on this register. After reset is removed, this field can be boot-loaded from the external non-volatile device (if present and valid) so that other legitimate PCI SIG members can substitute their vendor identification number for this field.

Figure 1. Vendor Identification Register

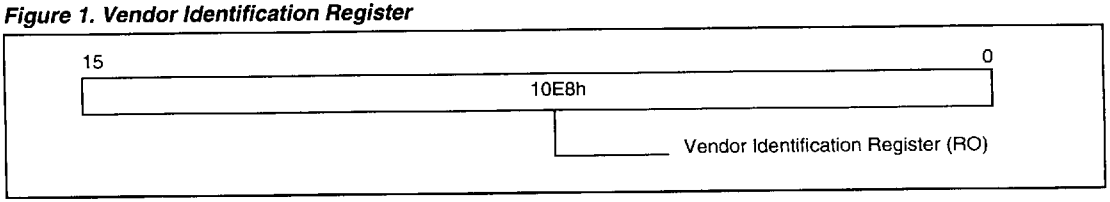


Table 2. Vendor Identification Register

Bit	Description
15:0	Vendor Identification Number: This is a 16 bit-value assigned to AMCC.

DEVICE IDENTIFICATION REGISTER (DID)

Register Name: Device Identification
Address Offset: 02h-03h
Power-up value: 4750h (ASCII hex for 'GP', General Purpose)
Boot-load: External nvRAM offset 042h-43h
Attribute: Read Only
Size: 16 bits

The DID register contains the vendor-assigned device identification number. This number is generated by AMCC in compliance with the conditions of the PCI specification. Write operations from the PCI interface have no effect on this register. After reset is removed, this field can be boot-loaded from the external non-volatile device (if present and valid) so that other legitimate PCI SIG members can substitute their own device identification number for this field.

Figure 2. Device Identification Register

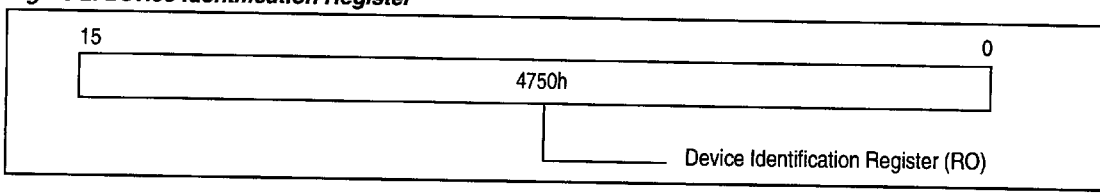


Table 3. Device Identification Register

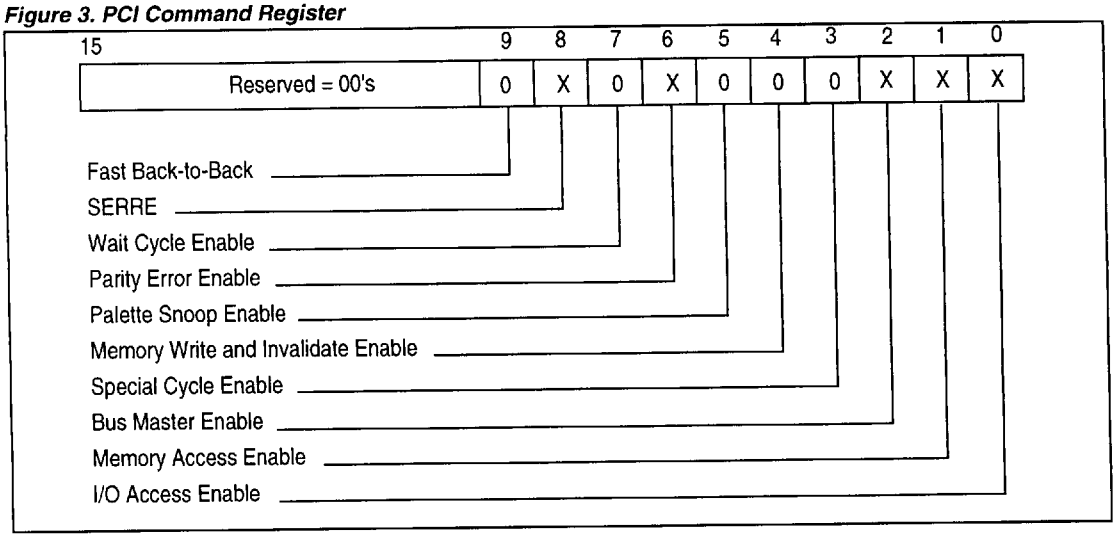
Bit	Description
15:0	Device Identification Number: This is a 16-bit value initially assigned by AMCC for applications using the AMCC Vendor ID.

PCI COMMAND REGISTER (PCICMD)

Register Name: PCI Command
Address Offset: 04h-05h
Power-up value: 0000h
Boot-load: not used
Attribute: Read/Write (R/W on 6 bits, Read Only for all others)
Size: 16 bits

This 16-bit register contains the PCI Command. The function of this register is defined by the PCI specification and its implementation is required of all PCI devices. Only six of the ten fields are used by this device; those which are not used are hardwired to 0. The definitions for all fields are provided here for completeness.

Figure 3. PCI Command Register



3

Table 4. PCI Command Register

Bit	Description
15:10	Reserved. Equals all 0's.
9	Fast Back-to-Back Enable. The S5933 does not support this function. This bit must be set to zero. This bit is cleared to a 0 upon RESET#.
8	System Error Enable. When this bit is set to 1, it permits the S5933 controller to drive the open drain output pin, SERR#. This bit is cleared to 0 upon RESET#. The SERR# pin driven active normally signifies a parity error on the address/control bus.
7	Wait Cycle Enable. This bit controls whether this device does address/data stepping. Since the S5933 controller never uses stepping, it is hardwired to 0.
6	Parity Error Enable. This bit, when set to a one, allows this controller to check for parity errors. When a parity error is detected, the PCI bus signal PERR# is asserted. This bit is cleared (parity testing disabled) upon the assertion of RESET#.
5	Palette Snoop Enable. This bit is not supported by the S5933 controller and is hardwired to 0. This feature is used solely for PCI-based VGA devices.
4	Memory Write and Invalidate Enable. This bit allows certain Bus Master devices to use the Memory Write and Invalidate PCI bus command when set to 1. When set to 0, masters must use the Memory Write command instead. The S5933 controller does not support this command when operated as a master and therefore it is hardwired to 0.
3	Special Cycle Enable. Devices which are capable of monitoring special cycles can do so when this bit is set to 1. The S5933 controller does not monitor (or generate) special cycles and this bit is hardwired to 0.
2	Bus Master Enable. This bit, when set to a one, allows the S5933 controller to function as a bus master. This bit is initialized to 0 upon the assertion of signal pin RESET#.
1	Memory Space Enable. This bit allows the S5933 controller to decode and respond as a target for memory regions that may be defined in one of the five base address registers. This bit is initialized to 0 upon the assertion of signal pin RESET#.
0	I/O Space Enable. This bit allows the S5933 controller to decode and respond as a target to I/O cycles which are to regions defined by any one of the five base address registers. This bit is initialized to 0 upon the assertion of signal pin RESET#.

PCI STATUS REGISTER (PCISTS)

Register Name: PCI Status

Address Offset: 06h-07h

Power-up value: 0080h

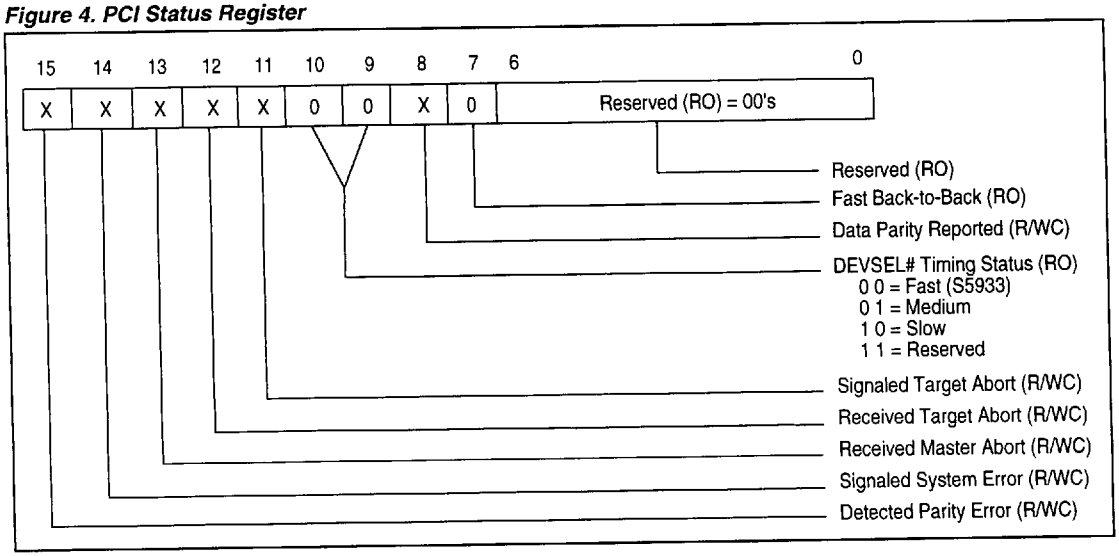
Boot-load: not used

Attribute: Read Only (RO), Read/Write Clear (R/WC)

Size: 16 bits

This 16-bit register contains the PCI status information. The function of this register is defined by the PCI specification and its implementation is required of all PCI devices. Only some of the bits are used by this device; those which are not used are hardwired to 0. Most status bits within this register are designated as "write clear," meaning that in order to clear a given bit, the bit must be written as a 1. All bits written with a 0 are left unchanged. These bits are identified in Figure 4 as (R/WC). Those which are Read Only are shown as (RO) in Figure 4.

Figure 4. PCI Status Register



3

Table 5. PCI Status Register

Bit	Description
15	Detected Parity Error. This bit is set whenever a parity error is detected. It functions independently from the state of Command Register Bit 6. This bit may be cleared by writing a 1 to this location.
14	Signaled System Error. This bit is set whenever the device asserts the signal SERR#. This bit can be reset by writing a 1 to this location.
13	Received Master Abort. This bit is set whenever a bus master abort occurs. This bit can be reset by writing a 1 to this location.
12	Received Target Abort. This bit is set whenever this device has one of its own initiated cycles terminated by the currently addressed target. This bit can be reset by writing a 1 to this location.
11	Signaled Target Abort. This bit is set whenever this device aborts a cycle when addressed as a target. This bit can be reset by writing a 1 to this location.
10:9	Device Select Timing. These bits are read-only and define the signal behavior of DEVSEL# from this device when accessed as a target.
8	Data Parity Reported. This bit is set upon the detection of a data parity error for a transfer involving the S5933 device as the master. The Parity Error Enable bit (D6 of the Command Register) must be set in order for this bit to be set. Once set, it can only be cleared by either writing a 1 to this location or by the assertion of the signal RESET#.
7	Fast Back-to-back Capable. When equal to 1, this indicates that the device can accept fast back-to-back cycles as a target.
6:0	Reserved. Equal all 0's.

REVISION IDENTIFICATION REGISTER (RID)

Register Name: Revision Identification
Address Offset: 08h
Power-up value: 00h
Boot-load: External nvRAM/EPROM offset 048h
Attribute: Read Only
Size: 8 bits

The RID register contains the revision identification number. This field is initially cleared. Write operations from the PCI interface have no effect on this register. After reset is removed, this field can be boot-loaded from the external non-volatile device (if present and valid) so that another value may be used.

Figure 5. Revision Identification Register

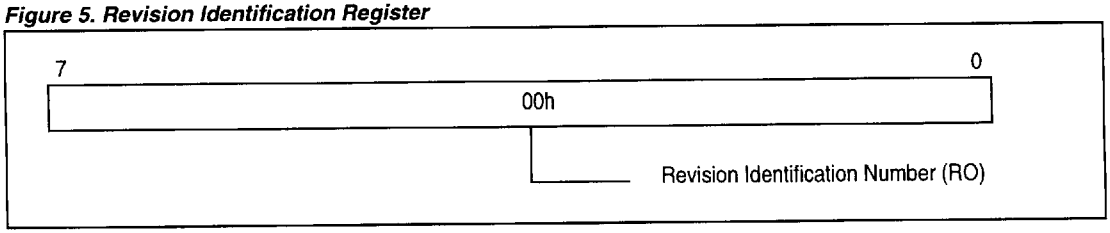


Table 6. Revision Identification Register

Bit	Description
7:0	Revision Identification Number. Initialized to zeros, this register may be loaded to the value in non-volatile memory at offset 048h.

CLASS CODE REGISTER (CLCD)

Register Name: Class Code
Address Offset: 09h-0Bh
Power-up value: FF0000h
Boot-load: External nvRAM offset 049h-4Bh
Attribute: Read Only
Size: 24 bits

This 24-bit, read-only register is divided into three one-byte fields: the base class resides at location 0Bh, the sub-class at 0Ah, and the programming interface at 09h. The default setting for the base class is all ones (FFh), which indicates that the device does not fit into the thirteen base classes defined in the PCI Local Bus Specification. It is possible, however, through use of the external non-volatile memory, to implement one of the defined class codes described in Table 7 below.

For devices that fall within the seven defined class codes, sub-classes are also assigned. Tables 8 through 20 describe each of the sub-class codes for base codes 00h through 0Ch, respectively.

Figure 6. Class Code Register

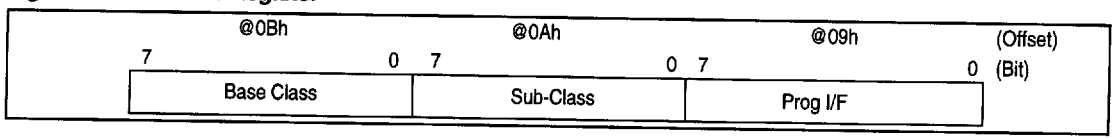


Table 7. Defined Base Class Codes

Base-Class	Description
00h	Early, pre-2.0 PCI specification devices
01h	Mass storage controller
02h	Network controller
03h	Display controller
04h	Multimedia device
05h	Memory controller
06h	Bridge device
07h	Simple communication controller
08h	Base system peripherals
09h	Input devices
0Ah	Docking stations
0Bh	Processors
0Ch	Serial bus controllers
0D-FEh	Reserved
FFh	Device does not fit defined class codes (default)

Table 8. Base Class Code 00h: Early, Pre-2.0 Specification Devices

Sub-Class	Prog I/F	Description
00h	00h	All devices other than VGA
01h	00h	VGA-compatible device

Table 9. Base Class Code 01h: Mass Storage Controllers

Sub-Class	Prog I/F	Description
00h	00h	SCSI controller
01h	xxh	IDE controller
02h	00h	Floppy disk controller
03h	00h	IPI controller
04h	00h	RAID controller
80h	00h	Other mass storage controller

Table 10. Base Class Code 02h: Network Controllers

Sub-Class	Prog I/F	Description
00h	00h	Ethernet controller
01h	00h	Token ring controller
02h	00h	FDDI controller
03h	00h	ATM controller
80h	00h	Other network controller

Table 11. Base Class Code 03h: Display Controllers

Sub-Class	Prog I/F	Description
00h	00h	VGA-compatible controller
00h	01h	8514 compatible controller
01h	00h	XGA controller
80h	00h	Other display controller

Table 12. Base Class Code 04h: Multimedia Devices

Sub-Class	Prog I/F	Description
00h	00h	Video device
01h	00h	Audio device
80h	00h	Other multimedia device

Table 13. Base Class Code 05h: Memory Controllers

Sub-Class	Prog I/F	Description
00h	00h	RAM memory controller
01h	00h	Flash memory controller
80h	00h	Other memory controller

Table 14. Base Class Code 06h: Bridge Devices

Sub-Class	Prog I/F	Description
00h	00h	Host/PCI bridge
01h	00h	PCI/ISA bridge
02h	00h	PCI/EISA bridge
03h	00h	PCI/Micro Channel bridge
04h	00h	PCI/PCI bridge
05h	00h	PCI/PCMCIA bridge
06h	00h	NuBus bridge
07h	00h	CardBus bridge
80h	00h	Other bridge type

Table 15. Base Class Code 07h: Simple Communications Controllers

Sub-Class	Prog I/F	Description
00h	00h	Generic XT compatible serial controller
	01h	16450 compatible serial controller
	02h	16550 compatible serial controller
01h	00h	Parallel port
	01h	Bidirectional parallel port
	02h	ECP 1.X compliant parallel port
80h	00h	Other communications device

Table 16. Base Class Code 08h: Base System Peripherals

Sub-Class	Prog I/F	Description
00h	00h	Generic 8259 PIC
	01h	ISA PIC
	02h	EISA PIC
01h	00h	Generic 8237 DMA controller
	01h	ISA DMA controller
	02h	EISA DMA controller
02h	00h	Generic 8254 system timer
	01h	ISA system timer
	02h	EISA system timers (2 timers)
03h	00h	Generic RTC controller
	01h	ISA RTC controller
80h	00h	Other system peripheral

Table 17. Base Class Code 09h: Input Devices

Sub-Class	Prog I/F	Description
00h	00h	Keyboard controller
01h	00h	Digitizer (Pen)
02h	00h	Mouse controller
80h	00h	Other input controller

Table 18. Base Class Code 0Ah: Docking Stations

Sub-Class	Prog I/F	Description
00h	00h	Generic docking station
80h	00h	Other type of docking station

Table 19. Base Class Code 0Bh: Processors

Sub-Class	Prog I/F	Description
00h	00h	Intel386™
01h	00h	Intel486™
02h	00h	Pentium™
10h	00h	Alpha™
40h	00h	Co-processor

Table 20. Base Class Code 0Ch: Serial Bus Controllers

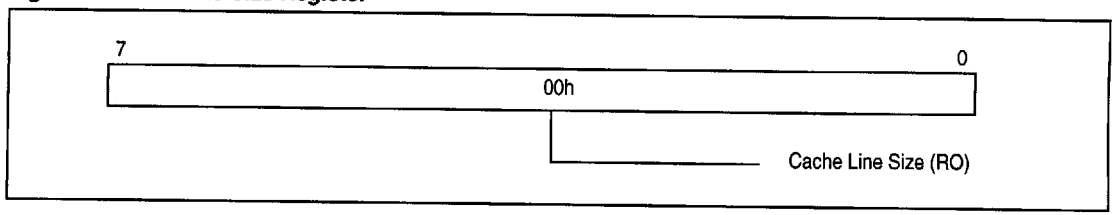
Sub-Class	Prog I/F	Description
00	00h	FireWire™ (IEEE 1394)
01h	00h	ACCESS.bus
02h	00h	SSA

CACHE LINE SIZE REGISTER (CALN)

Register Name:	Cache Line Size
Address Offset:	0Ch
Power-up value:	00h, hardwired
Boot-load:	not used
Attribute:	Read Only
Size:	8 bits

This register is hardwired to 0. The cache line configuration register is used by the system to define the cache line size in doubleword (64-bit) increments. This controller does not use the "Memory Write and Invalidate" PCI bus cycle commands when operating in the bus master mode, and therefore does not internally require this register. When operating in the target mode, this controller does not have the connections necessary to "snoop" the PCI bus and accordingly cannot employ this register in the detection of burst transfers that cross a line boundary.

Figure 7. Cache Line Size Register

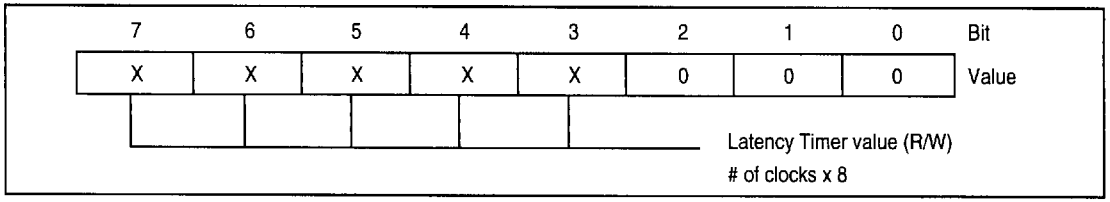


LATENCY TIMER REGISTER (LAT)

Register Name:	Latency Timer
Address Offset:	0Dh
Power-up value:	00h
Boot-load:	External nvRAM offset 04Dh
Attribute:	Read/Write, bits 7:3; Read Only bits 2:0
Size:	8 bits

The latency timer register has meaning only when this controller is used as a bus master and pertains to the number of PCI bus clocks that this master will be guaranteed. The nonzero value for this register is internally decremented after this device has been granted the bus and has begun to assert FRAME#. Prior to this latency timer count reaching zero, this device can ignore the removal of the bus grant and may continue the use of the bus for data transfers.

Figure 8. Latency Timer Register



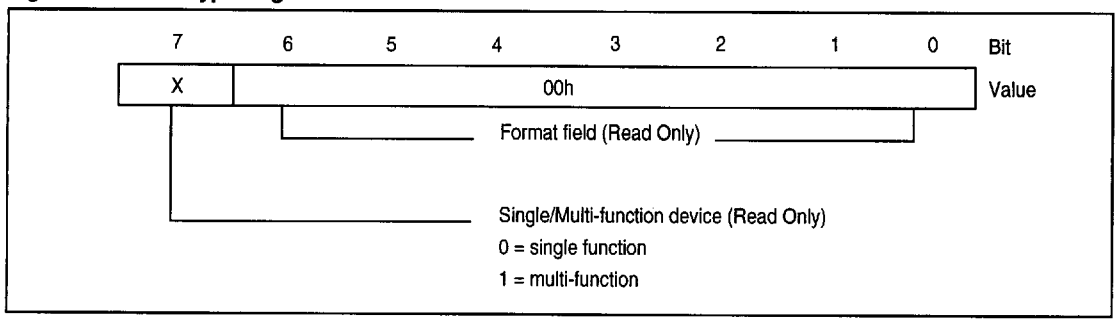
3

HEADER TYPE REGISTER (HDR)

Register Name: Header Type
Address Offset: 0Eh
Power-up value: 00h
Boot-load: External nvRAM offset 04Eh
Attribute: Read Only
Size: 8 bits

This register consists of two fields: Bits 6:0 define the format for bytes 10h through 3Fh of the device configuration header, and bit 7 establishes whether this device represents a single function (bit 7 = 0) or a multifunction (bit 7 = 1) PCI bus agent. The S5933 is a single function PCI device.

Figure 9. Header Type Register



BUILT-IN SELF-TEST REGISTER (BIST)

Register Name: Built-in Self-Test
Address Offset: 0Fh
Power-up value: 00h
Boot-load: External nvRAM/EPROM offset 04Fh
Attribute: D7, D5-0 Read Only, D6 as PCI bus write only
Size: 8 bits

The Built-In Self-Test (BIST) register permits the implementation of custom, user-specific diagnostics. This register has four fields as depicted in Figure 10. Bit 7, when set signifies that this device supports a built-in self test. When bit 7 is set, writing a 1 to bit 6 will commence the self test. In actuality, writing a 1 to bit 6 produces an interrupt to the Add-On interface. Bit 6 will remain set until cleared by a write operation to this register from the Add-On bus interface. When bit 6 is reset it is interpreted as completion of the self-test and an error is indicated by a non-zero value for the completion code (bits 3:0).

Figure 10. Built-In Self Test Register

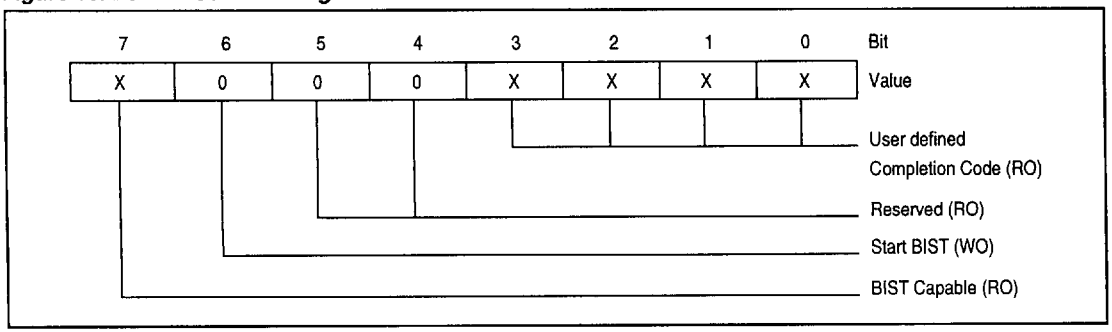


Table 21. Built-In Self-Test Register

Bit	Description
7	BIST Capable. This bit indicates that the Add-On device supports a built-in self-test when a one is returned. A zero should be returned if this self test feature is not desired. This field is read only from the PCI interface.
6	Start BIST. Writing a 1 to this bit indicates that the self-test should commence. This bit can only be written when bit 7 is a 1. When bit 6 becomes set, an interrupt is issued to the Add-On interface. Other than through the reset pin, Bit 6 can only be cleared by a write to this element from the Add-On bus interface as outlined in Section 6.5. The PCI bus specification requires that this bit be cleared within 2 seconds after being set, or the device will be failed.
5:4	Reserved. These bits are reserved. This field will always return zeros.
3:0	Completion Code. This field provides a method for detailing a device-specific error. It is considered valid when the Start BIST field (bit 6) changes from 1 to 0. An all-zero value for the completion code indicates successful completion.

BASE ADDRESS REGISTERS (BADR)

Register Name:	Base Address
Address Offset:	10h, 14h, 18h, 1Ch, 20h, 24h
Power-up value:	FFFFFFC1h for offset 10h; 00000000h for all others
Boot-load:	External nvRAM offset 050h, 54h, 58h, 5Ch, 60h (BADR0-4)
Attribute:	high bits Read/Write; low bits Read Only
Size:	32 bits

The base address registers provide a mechanism for assigning memory or I/O space for the Add-On function. The actual location(s) the Add-On function is to respond to is determined by first interrogating these registers to ascertain the size or space desired, and then writing the high-order field of each register to place it physically in the system's address space. Bit zero of each field is used to select whether the space required is to be decoded as memory (bit 0 = 0) or I/O (bit 0 = 1). Since this PCI controller has 16 DWORDs of internal operating registers, the Base Address Register at offset 10h is assigned to them. The remaining five base address registers can only be used by boot-loading them from the external nvRAM interface. BADR5 register is not implemented and will return all 0's.

Determining Base Address Size

The address space defined by a given base address register is determined by writing all 1s to a given base address register from the PCI bus and then reading that register back. The number of 0s returned starting from D4 for memory space and D2 for I/O space toward the high-order bits reveals the amount of address space desired. Tables 23 and 24 list the possible returned values and their corresponding size for both memory and I/O, respectively. Included in the table are the nvRAM/EPROM boot values which correspond to a given assigned size. A register returning all zeros is disabled.

Assigning the Base Address

After a base address has been sized as described in the preceding paragraph, the region associated with that base address register (the high order one bits) can physically locate it in memory (or I/O) space. For example, the first base address register returns FFFFFFC1h indicating an I/O space (D0=1) and is then written with the value 00000300h. This means that the controller's internal registers can be selected for I/O addresses between 00000300h through 0000033Fh, in this example. The base address value must be on a natural binary boundary for the required size (example 300h, 340h, 380h etc.; 338h would not be allowable).

Figure 11a. Base Address Register — Memory

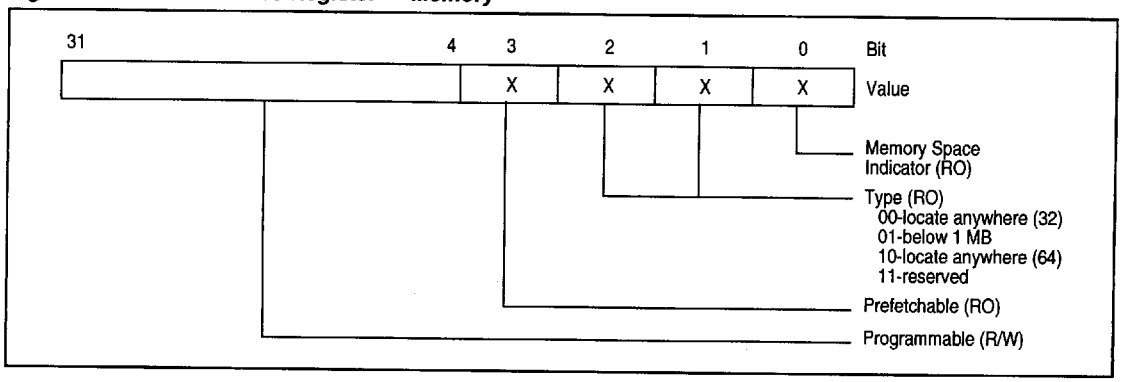


Figure 11b. Base Address Register — I/O

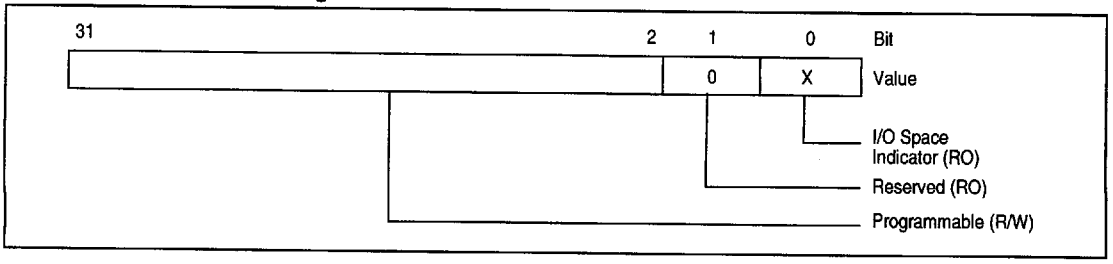


Table 22a. Base Address Register — Memory (Bit 0 = 0)

Bit	Description
31:4	Base Address Location. These bits are used to position the decoded region in memory space. Only bits which return a 1 after being written as 1 are usable for this purpose. Except for Base Address Register 0, these bits are individually enabled by the contents sourced from the external boot memory.
3	Prefetchable. When set as a 1, this bit signifies that this region of memory can be cached. Cachable regions can only be located within the region altered through PCI bus memory writes. This bit, when set, also implies that all read operations will return the data associated for all bytes regardless of the Byte Enables. Memory space which cannot support this behavior should leave this bit in the zero state. For Base Addresses 1 through 4, this bit is set by the Reset pin and later initialized by the external boot memory (if present). Base Address Register 0 always has this bit set to 0. This bit is read only from the PCI interface.
2:1	Memory Type. These two bits identify whether the memory space is 32 or 64 bits wide and if the space location is restricted to be within the first megabyte of memory space. The table below describes the encoding: **Bits 2 1** — **Description** 0 0 — Region is 32 bits wide and can be located anywhere in 32 bit memory space. 0 1 — Region is 32 bits wide and must be mapped below the first MByte of memory space. 1 0 — Region is 64 bits wide and can be mapped anywhere within 64 bit memory space. (Not supported by this controller.) 1 1 — Reserved. (Not supported by this controller.)
1	The 64-bit memory space is not supported by this controller, so bit 2 should not be set. The only meaningful option is whether it is desired to position memory space anywhere within 32-bit memory space or restrain it to the first megabyte. For Base Addresses 1 through 5, this bit is set by the reset pin and later initialized by the external boot memory (if present).
0	Space Indicator = 0. When set to 0, this bit identifies a base address region as a memory space and the remaining bits in the base address register are defined as shown in Table 22a.

Table 22b. Base Address Register — I/O (Bit 0 = 1)

Bit	Description
31:2	Base Address Location. These bits are used to position the decoded region in I/O space. Only bits which return a "1" after being written as "1" are usable for this purpose. Except for Base Address 0, these bits are individually enabled by the contents sourced from the external boot memory (EPROM or nvRAM).
1	Reserved. This bit should be zero. (Note: disabled Base Address Registers will return all zeros for the entire register location, bits 31 through 0).
0	Space Indicator = 1. When one this bit identifies a base address region as an I/O space and the remaining bits in the base address register have the definition as shown in Table 11b.

Table 23. Read Response (Memory Assigned) to an All-Ones Write Operation to a Base Address Register

Response	Size in bytes	[EPROM boot value] [1]
00000000h	none - disabled	00000000h or BIOS missing [2,3]
FFFFFFF0h	16 bytes (4 DWORDs)	FFFFFFF0h
FFFFFFE0h	32 bytes (8 DWORDs)	FFFFFFE0h
FFFFFFC0h	64 bytes (16 DWORDs)	FFFFFFC0h
FFFFFF80h	128 bytes (32 DWORDs)	FFFFFF80h
FFFFFF00h	256 bytes (64 DWORDs)	FFFFFF00h
FFFFFE00h	512 bytes (128 DWORDs)	FFFFFE00h
FFFFFC00h	1K bytes (256 DWORDs)	FFFFFC00h
FFFFF800h	2K bytes (512 DWORDs)	FFFFF800h
FFFFF000h	4K bytes (1K DWORDs)	FFFFF000h
FFFFE000h	8K bytes (2K DWORDs)	FFFFE000h
FFFFC000h	16K bytes (4K DWORDs)	FFFFC000h
FFFF8000h	32K bytes (8K DWORDs)	FFFF8000h
FFFF0000h	64K bytes (16K DWORDs)	FFFF0000h
FFFE0000h	128K bytes (32K DWORDs)	FFFE0000h
FFFC0000h	256K bytes (64K DWORDs)	FFFC0000h
FFF80000h	512K bytes (128K DWORDs)	FFF80000h
FFF00000h	1M bytes (256K DWORDs)	FFF00000h
FFE00000h	2M bytes (512K DWORDs)	FFE00000h
FFC00000h	4M bytes (1M DWORDs)	FFC00000h
FF800000h	8M bytes (2M DWORDs)	FF800000h
FF000000h	16M bytes (4M DWORDs)	FF000000h
FE000000h	32M bytes (8M DWORDs)	FE000000h
FC000000h	64M bytes (16M DWORDs)	FC000000h
F8000000h	128M bytes (32M DWORDs)	F8000000h
F0000000h	256M bytes (64M DWORDs)	F0000000h
E0000000h	512M bytes (128M DWORDs)	E0000000h

1. The two most significant bits define bus width for BADR1:4 in Pass-Thru operation).
2. Bits D3, D2 and D1 may be set to indicate other attributes for the memory space. See text for details.
3. BADR5 register is not implemented and will return all 0's.

Table 24. Read Response (I/O Assigned) to an All-Ones write Operation to a Base Address Register

Response	Size in bytes	[EPROM boot value]
00000000h	none - disabled	00000000h or BIOS missing [3]
FFFFFFFDh	4 bytes (1 DWORDs)	FFFFFFFDh
FFFFFFF9h	8 bytes (2 DWORDs)	FFFFFFF9h
FFFFFFF1h	16 bytes (4 DWORDs)	FFFFFFF1h
FFFFFFE1h	32 bytes (8 DWORDs)	FFFFFFE1h
FFFFFFC1h	64 bytes (16 DWORDs)	FFFFFFC1h [4]
FFFFFF81h	128 bytes (32 DWORDs)	FFFFFF81h
FFFFFF01h	256 bytes (64 DWORDs)	FFFFFF01h

4. Base Address Register 0 (at offset) 10h powers up as FFFFFFC1h. This default assignment allows usage without an external boot memory. Should an EPROM or nvRAM be used, the base address can be boot loaded to become a memory space (FFFFFFC0h or FFFFFFC2h).

3

EXPANSION ROM BASE ADDRESS REGISTER (XROM)

Register Name:	Expansion ROM Base Address
Address Offset:	30h
Power-up value:	00000000h
Boot-load:	External nvRAM offset 70h
Attribute:	bits 31:11, bit 0 Read/Write; bits 10:1 Read Only
Size:	32 bits

The expansion base address ROM register provides a mechanism for assigning a space within physical memory for an expansion ROM. Access from the PCI bus to the memory space defined by this register will cause one or more accesses to the S5933 controllers' external BIOS ROM (or nvRAM) interface. Since PCI bus accesses to the ROM may be 32 bits wide, repeated operations to the ROM are generated by the S5933 and the wider data is assembled internal to the S5933 controller and then transferred to the PCI bus by the S5933.

Figure 12. Expansion ROM Base Address Register

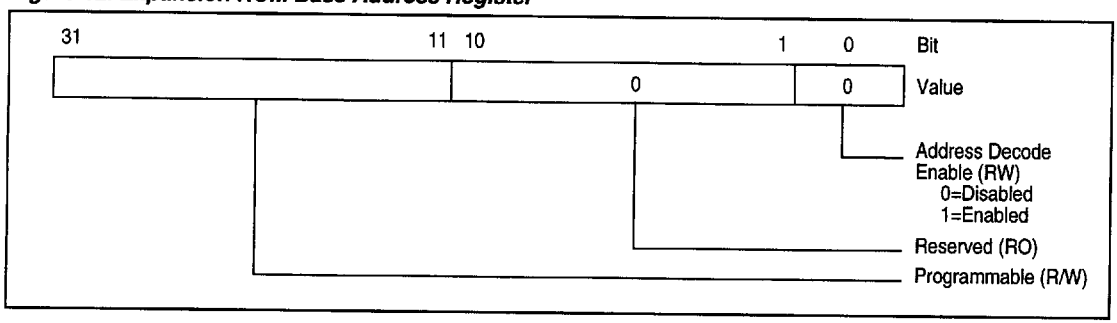


Table 25. Expansion ROM Base Address Register

Bit	Description
31:11	Expansion ROM Base Address Location. These bits are used to position the decoded region in memory space. Only bits which return a 1 after being written as 1 are usable for this purpose. These bits are individually enabled by the contents sourced from the external boot memory (EPROM or nvRAM). The desired size for the ROM memory is determined by writing all ones to this register and then reading back the contents. The number of bits returned as zeros, in order from least significant to most significant bit, indicates the size of the expansion ROM. This controller limits the expansion ROM area to 64K bytes. The allowable returned values after all ones are written to this register are shown in Table 26.
10:1	Reserved. All zeros.
0	Address Decode Enable. The Expansion ROM address decoder is enabled or disabled with this bit. When this bit is set, the decoder is enabled; when this bit is zero, the decoder is disabled. It is required that the PCI command register also have the memory decode enabled for this bit to have an effect.

Table 26. Read Response to Expansion ROM Base Address Register (after all-ones written)

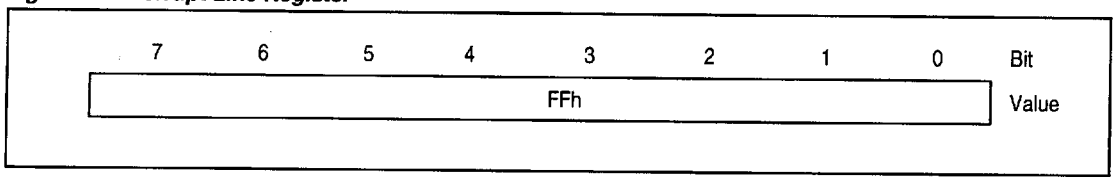
Response	Size in bytes	[EPROM boot value]
00000000h	none - disabled	00000000h or BIOS missing
FFFFF801h	2K bytes (512 DWORDs)	FFFFF801h
FFFFF001h	4K bytes (1K DWORDs)	FFFFF001h
FFFFE001h	8K bytes (2K DWORDs)	FFFFE001h
FFFFC001h	16K bytes (4K DWORDs)	FFFFC001h
FFFF8001h	32K bytes (8K DWORDs)	FFFF8001h
FFFF0001h	64K bytes (16K DWORDs)	FFFF0001h

INTERRUPT LINE REGISTER (INTLN)

Register Name:	Interrupt Line
Address Offset:	3Ch
Power-up value:	FFh
Boot-load:	External nvRAM offset 7Ch
Attribute:	Read/Write
Size:	8 bit

This register indicates the interrupt routing for the S5933 controller. The ultimate value for this register is system-architecture specific. For x86 based PCs, the values in this register correspond with the established interrupt numbers associated with the dual 8259 controllers used in those machines. In x86-based PC systems, the values of 0 to 15 correspond with the IRQ numbers 0 through 15, and the values from 16 to 254 are reserved. The value of 255 (the controller's default power-up value) signifies either "unknown" or "no connection" for the system interrupt. This register is boot-loaded from the external boot memory, if present, and may be written by the PCI interface.

Figure 13. Interrupt Line Register

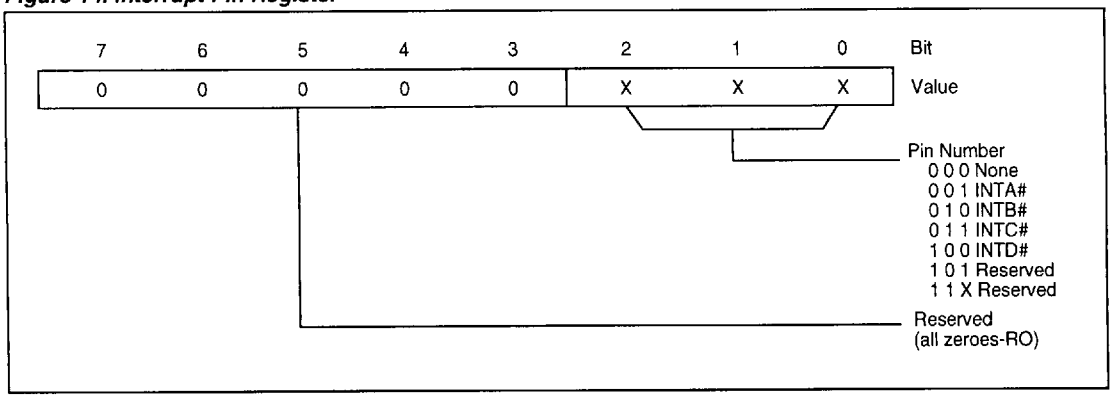


INTERRUPT PIN REGISTER (INTPIN)

Register Name: Interrupt Pin
Address Offset: 3Dh
Power-up value: 01h
Boot-load: External nvRAM offset 7Dh
Attribute: Read Only
Size: 8 bits

This register identifies which PCI interrupt, if any, is connected to the controller's PCI interrupt pins. The allowable values are 0 (no interrupts), 1 (INTA#), 2 (INTB#), 3 (INTC#), and 4 (INTD#). The default power-up value assumes INTA#.

Figure 14. Interrupt Pin Register



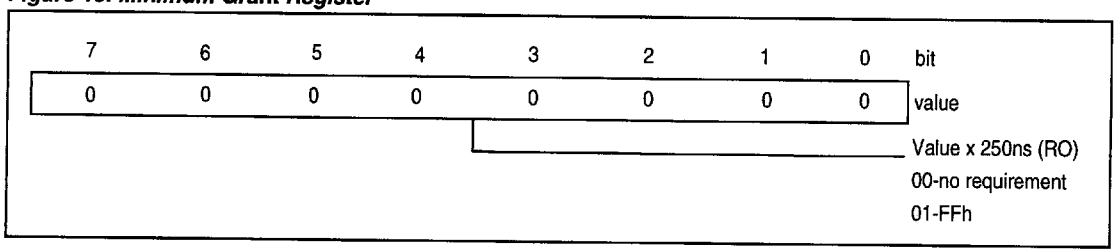
MINIMUM GRANT REGISTER (MINGNT)

Register Name:	Minimum Grant
Address Offset:	3Eh
Power-up value:	00h
Boot-load:	External nvRAM offset 7Eh
Attribute:	Read Only
Size:	8 bits

This register may be optionally used by bus masters to specify how long a burst period the device needs. A value of zero indicates that the bus master has no stringent requirement. The units defined by the least significant bit are in 250-ns increments. This register is treated as "information only" and has no further implementation within this device.

Values other than zero are possible when an external boot memory is used.

Figure 15. Minimum Grant Register



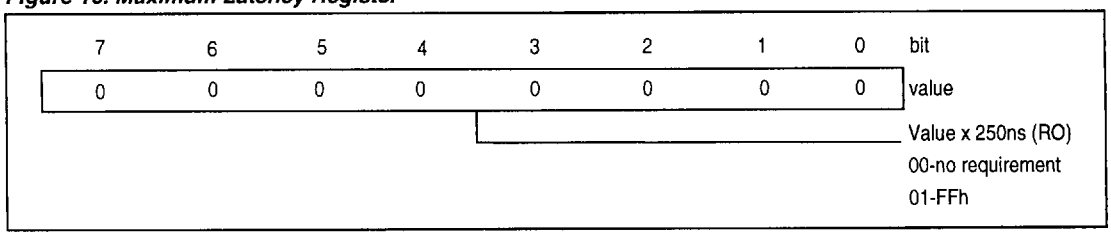
MAXIMUM LATENCY REGISTER (MAXLAT)

Register Name:	Maximum Latency
Address Offset:	3Fh
Power-up value:	00h
Boot-load:	External nvRAM offset 7Fh
Attribute:	Read Only
Size:	8 bits

This register may be optionally used by bus masters to specify how often this device needs PCI bus access. A value of zero indicates that the bus master has no stringent requirement. The units defined by the least significant bit are in 250-ns increments. This register is treated as "information only" and has no further implementation within this device.

Values other than zero are possible when an external boot memory is used.

Figure 16. Maximum Latency Register



3

PAGE(S) INTENTIONALLY BLANK

PCI BUS OPERATION REGISTERS

The PCI bus operation registers are mapped as 16 consecutive DWORD registers located at the address space (I/O or memory) specified by the Base Address Register 0. These locations are the primary method of communication between the PCI and Add-On buses. Data, software-defined commands and command parameters can be either exchanged through the mailboxes, transferred through the FIFO in blocks under program control, or transferred using the FIFOs under Bus Master control. Table 1 lists the PCI Bus Operation Registers.

Table 1. Operation Registers — PCI Bus

Address Offset	Abbreviation	Register Name
00h	OMB1	Outgoing Mailbox Register 1
04h	OMB2	Outgoing Mailbox Register 2
08h	OMB3	Outgoing Mailbox Register 3
0Ch	OMB4	Outgoing Mailbox Register 4
10h	IMB1	Incoming Mailbox Register 1
14h	IMB2	Incoming Mailbox Register 2
18h	IMB3	Incoming Mailbox Register 3
1Ch	IMB4	Incoming Mailbox Register 4
20h	FIFO	FIFO Register port (bidirectional)
24h	MWAR	Master Write Address Register
28h	MWTC	Master Write Transfer Count Register
2Ch	MRAR	Master Read Address Register
30h	MRTC	Master Read Transfer Count Register
34h	MBEF	Mailbox Empty/Full Status
38h	INTCSR	Interrupt Control/Status Register
3Ch	MCSR	Bus Master Control/Status Register

OUTGOING MAILBOX REGISTERS (OMB)

Register Names: Outgoing Mailboxes 1-4

PCI Address Offset: 00h, 04h, 08h, 0Ch

Power-up value: XXXXXXXXh

Attribute: Read/Write

Size: 32 bits

These four DWORD registers provide a method for sending command or parameter data to the Add-On system. PCI bus operations to these registers may be in any width (byte, word, or DWORD). Writing to these registers can be a source for Add-On bus interrupts (if desired) by enabling their interrupt generation through the use of the Add-On's interrupt control/status register.

INCOMING MAILBOX REGISTERS (IMB)

Register Names: Incoming Mailboxes 1-4

PCI Address Offset: 10h, 14h, 18h, 1Ch

Power-up value: XXXXXXXXh

Attribute: Read Only

Size: 32 bits

These four DWORD registers provide a method for receiving user defined data from the Add-On system. PCI bus read operations to these registers may be in any width (byte, word, or DWORD). Only read operations are supported. Reading from these registers can optionally cause an Add-On bus interrupt (if desired) by enabling their interrupt generation through the use of the Add-On's interrupt control/status register.

Mailbox 4, byte 3 only exists as device pins on the S5933 devices when used with a serial nonvolatile memory.

FIFO REGISTER PORT (FIFO)

Register Name: FIFO Port

PCI Address Offset: 20h

Power-up value: XXXXXXXXh

Attribute: Read/Write

Size: 32 bits

This location provides access to the bidirectional FIFO. Separate registers are used when reading from or writing to the FIFO. Accordingly, it is not possible to read what was written to this location. The FIFO registers are implicitly involved in all bus master operations and, as such, should not be accessed during active bus master transfers. When operating upon the FIFOs with software program transfers involving word or byte operations, the *endian* sequence of the FIFO should be established as described under FIFO Endian Conversion Management in order to preserve the internal FIFO data ordering and flag management. The FIFO's fullness may be observed by reading the master control- status registeror MCSR register.

PCI CONTROLLED BUS MASTER WRITE ADDRESS REGISTER (MWAR)

Register Name: Master Write Address
PCI Address Offset: 24h
Power-up value: 00000000h
Attribute: Read/Write
Size: 32 bits

This register is used to establish the PCI address for data moving from the Add-On bus to the PCI bus during PCI bus memory write operations. It consists of a 30-bit counter with the low-order two bits hardwired as zeros. Transfers may be any non-zero byte length as defined by the transfer count register, MWTC, and must begin on a DWORD boundary. This DWORD boundary starting constraint is placed upon this controller's PCI bus master transfers so that byte lane alignment can be maintained between the S5933 controller's internal FIFO data path, the Add-On interface, and the PCI bus.

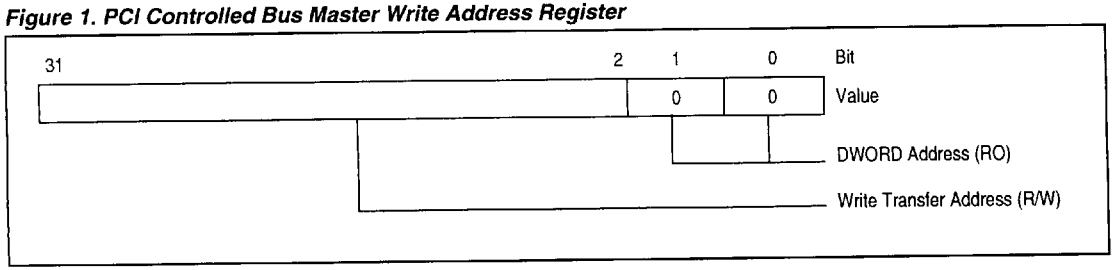
Note: Applications which require a non-DWORD starting boundary will need to move the first few bytes under software program control (and without using the FIFO) to establish a DWORD boundary. After the DWORD boundary is established the S5933 can begin the task of PCI bus master data transfers.

The Master Write Address Register is continually updated during the transfer process and will always be pointing to the next unwritten location. Reading of this register during a transfer process (done when the S5933 controller is functioning as a target, i.e. not a bus master) is permitted and may be used to monitor the progress of the transfer. During the address phase for bus master write transfers, the two least significant bits presented on the PCI bus pins AD[31:0] will always be zero. This identifies to the target memory that the burst address sequence will be in a linear order rather than in an Intel 486 or Pentium™ cache line fill sequence. Also, the PCI bus address bit A1 will always be zero when this controller is the bus master. This signifies to the target that the S5933 controller is burst capable and that the target should not arbitrarily disconnect after the first data phase of this operation.

Under certain circumstances, MWAR can be accessed from the Add-On bus instead of the PCI bus. See Add-On Initiated Bus Mastering.

3

Figure 1. PCI Controlled Bus Master Write Address Register



PCI CONTROLLED BUS MASTER WRITE TRANSFER COUNT REGISTER (MWTC)

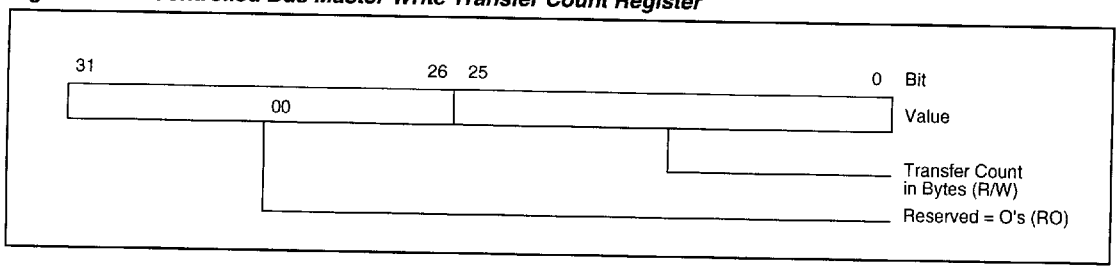
Register Name: Master Write Transfer Count
PCI Address Offset: 28h
Power-up value: 00000000h
Attribute: Read/Write
Size: 32 bits

The master write transfer count register is used to convey to the S5933 controller the actual number of bytes that are to be transferred. The value in this register is decremented with each bus master PCI write operation until the transfer count reaches zero.

Upon reaching zero, the transfer operation ceases and an interrupt may be optionally generated to either the PCI or Add-On bus interface. Transfers which are not whole multiples of DWORDs in size result in a partial word ending cycle. This partial word ending cycle is possible since all bus master transfers for this controller are required to begin on a DWORD boundary.

Under certain circumstances, MWTC can be accessed from the Add-On bus instead of the PCI bus. See Add-On Initiated Bus Mastering.

Figure 2. PCI Controlled Bus Master Write Transfer Count Register



PCI CONTROLLED BUS MASTER READ ADDRESS REGISTER (MRAR)

Register Name: Master Read Address
PCI Address Offset: 2Ch
Power-up value: 00000000h
Attribute: Read/Write
Size: 32 bits

This register is used to establish the PCI address for data moving to the Add-On bus from the PCI bus during PCI bus memory read operations. It consists of a 30-bit counter with the low-order two bits hardwired as zeros. Transfers may be any non-zero byte length as defined by the transfer count register, MRTC (Section 5.7) and must begin on a DWORD boundary. This DWORD boundary starting constraint is placed upon this controller's PCI bus master transfers so that byte lane alignment can be maintained between the S5933 controller's internal FIFO data path, the Add-On interface and the PCI bus.

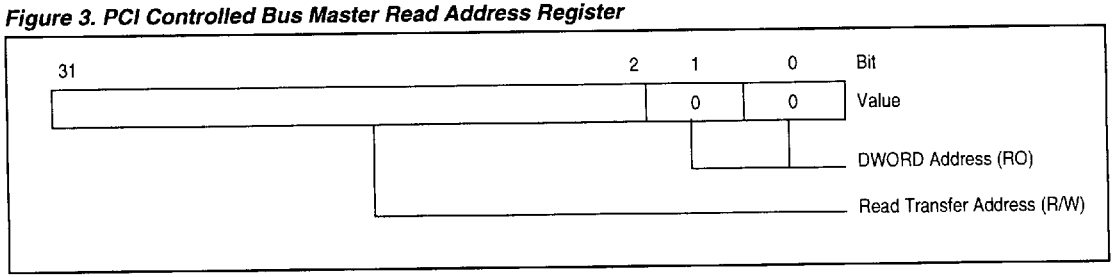
Note: Applications which require a non-DWORD starting boundary will need to move the first few bytes under software program control (and without using the FIFO) to establish a DWORD boundary. After the DWORD boundary is established the S5933 can begin the task of PCI bus master data transfers.

The Master Read Address Register is continually updated during the transfer process and will always be pointing to the next unread location. Reading of this register during a transfer process (done when the S5933 controller is functioning as a target—i.e., not a bus master) is permitted and may be used to monitor the progress of the transfer. During the address phase for bus master read transfers, the two least significant bits presented on the PCI bus AD[31:0] will always be zero. This identifies to the target memory that the burst address sequence will be in a linear order rather than in an Intel 486 or Pentium™ cache line fill sequence. Also, the PCI bus address bit A1 will always be zero when this controller is the bus master. This signifies to the target that the controller is burst capable and that the target should not arbitrarily disconnect after the first data phase of this operation.

Under certain circumstances, MRAR can be accessed from the Add-On bus instead of the PCI bus.

3

Figure 3. PCI Controlled Bus Master Read Address Register



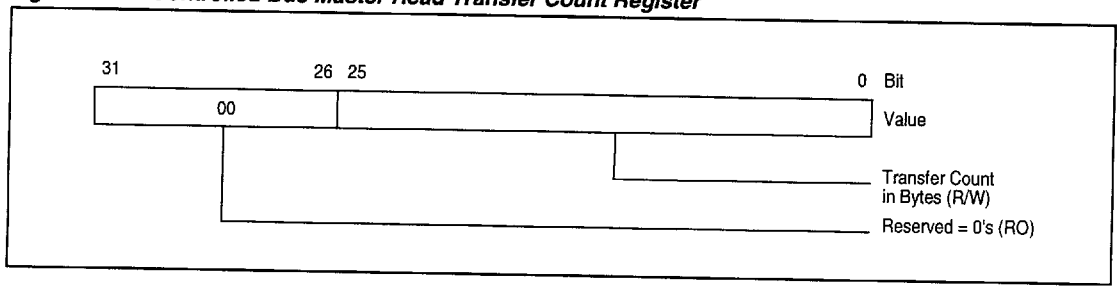
PCI CONTROLLED BUS MASTER READ TRANSFER COUNT REGISTER (MRTC)

Register Name: Master Read Transfer Count
PCI Address Offset: 30h
Power-up value: 00000000h
Attribute: Read/Write
Size: 32 bits

The master read transfer count register is used to convey to the PCI controller the actual number of bytes that are to be transferred. The value in this register is decremented with each bus master PCI read operation until the transfer count reaches zero. Upon reaching zero, the transfer operation ceases and an interrupt may be optionally generated to either the PCI or Add-On bus interface. Transfers which are not whole multiples of DWORDs in size result in a partial word ending cycle. This partial word ending cycle is possible since all bus master transfers for this controller are required to begin on a DWORD boundary.

Under certain circumstances, MRTC can be accessed from the Add-On bus instead of the PCI bus.

Figure 4. PCI Controlled Bus Master Read Transfer Count Register



MAILBOX EMPTY FULL/STATUS REGISTER (MBEF)

Register Name: Mailbox Empty/Full Status

PCI Address Offset: 34h

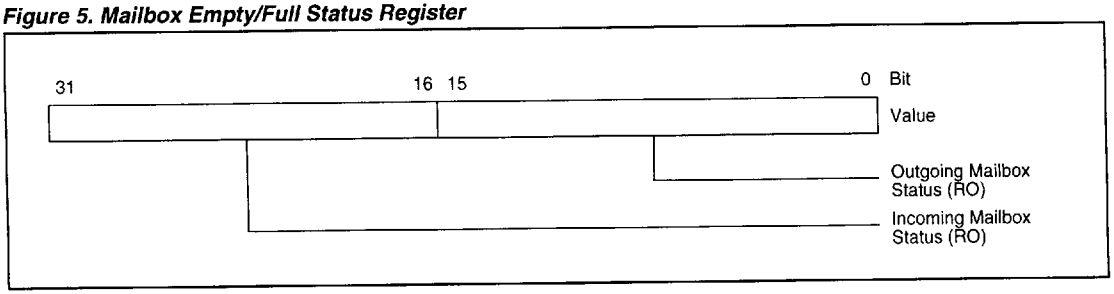
Power-up value: 00000000h

Attribute: Read Only

Size: 32 bits

This register provides empty/full visibility of each byte within the mailboxes. The empty/full status for the Outgoing mailboxes is displayed on the low-order 16 bits and the empty/full status for the Incoming mailboxes is presented on the high-order 16 bits. A value of 1 signifies that a given mailbox has been written by one bus interface but has not yet been read by the corresponding destination interface. A PCI bus incoming mailbox is defined as one in which data travels from the Add-On bus into the PCI bus, and an outgoing mailbox is defined as one where data travels out from the PCI bus to the Add-On interface.

Figure 5. Mailbox Empty/Full Status Register



3

Table 2. Mailbox Empty/Full Status Register

Bit	Description
31:16	Incoming Mailbox Status. This field indicates which incoming mailbox registers have been written by the Add-On interface but have not yet been read by the PCI bus. Each bit location corresponds to a specific byte within one of the four incoming mailboxes. A value of one for each bit signifies that the specified mailbox byte is full, and a value of zero signifies empty. The mapping of these status bits to bytes within each mailbox is as follows: Bit 31 = Incoming mailbox 4 byte 3 Bit 30 = Incoming mailbox 4 byte 2 Bit 29 = Incoming mailbox 4 byte 1 Bit 28 = Incoming mailbox 4 byte 0 Bit 27 = Incoming mailbox 3 byte 3 Bit 26 = Incoming mailbox 3 byte 2 Bit 25 = Incoming mailbox 3 byte 1 Bit 24 = Incoming mailbox 3 byte 0 Bit 23 = Incoming mailbox 2 byte 3 Bit 22 = Incoming mailbox 2 byte 2 Bit 21 = Incoming mailbox 2 byte 1 Bit 20 = Incoming mailbox 2 byte 0 Bit 19 = Incoming mailbox 1 byte 3 Bit 18 = Incoming mailbox 1 byte 2 Bit 17 = Incoming mailbox 1 byte 1 Bit 16 = Incoming mailbox 1 byte 0
15:00	Outgoing Mailbox Status. This field indicates which out going mail box registers have been written by the PCI bus interface but have not yet been read by the Add-On bus. Each bit location corresponds to a specific byte within one of the four outgoing mailboxes. A value of one for each bit signifies that the specified mailbox byte is full, and a value of zero signifies empty. The mapping of these status bits to bytes within each mailbox is as follows: Bit 15 = Outgoing mailbox 4 byte 3 Bit 14 = Outgoing mailbox 4 byte 2 Bit 13 = Outgoing mailbox 4 byte 1 Bit 12 = Outgoing mailbox 4 byte 0 Bit 11 = Outgoing mailbox 3 byte 3 Bit 10 = Outgoing mailbox 3 byte 2 Bit 09 = Outgoing mailbox 3 byte 1 Bit 08 = Outgoing mailbox 3 byte 0 Bit 07 = Outgoing mailbox 2 byte 3 Bit 06 = Outgoing mailbox 2 byte 2 Bit 05 = Outgoing mailbox 2 byte 1 Bit 04 = Outgoing Mailbox 2 byte 0 Bit 03 = Outgoing Mailbox 1 byte 3 Bit 02 = Outgoing Mailbox 1 byte 2 Bit 01 = Outgoing Mailbox 1 byte 1 Bit 00 = Outgoing Mailbox 1 byte 0

INTERRUPT CONTROL/STATUS REGISTER (INTCSR)

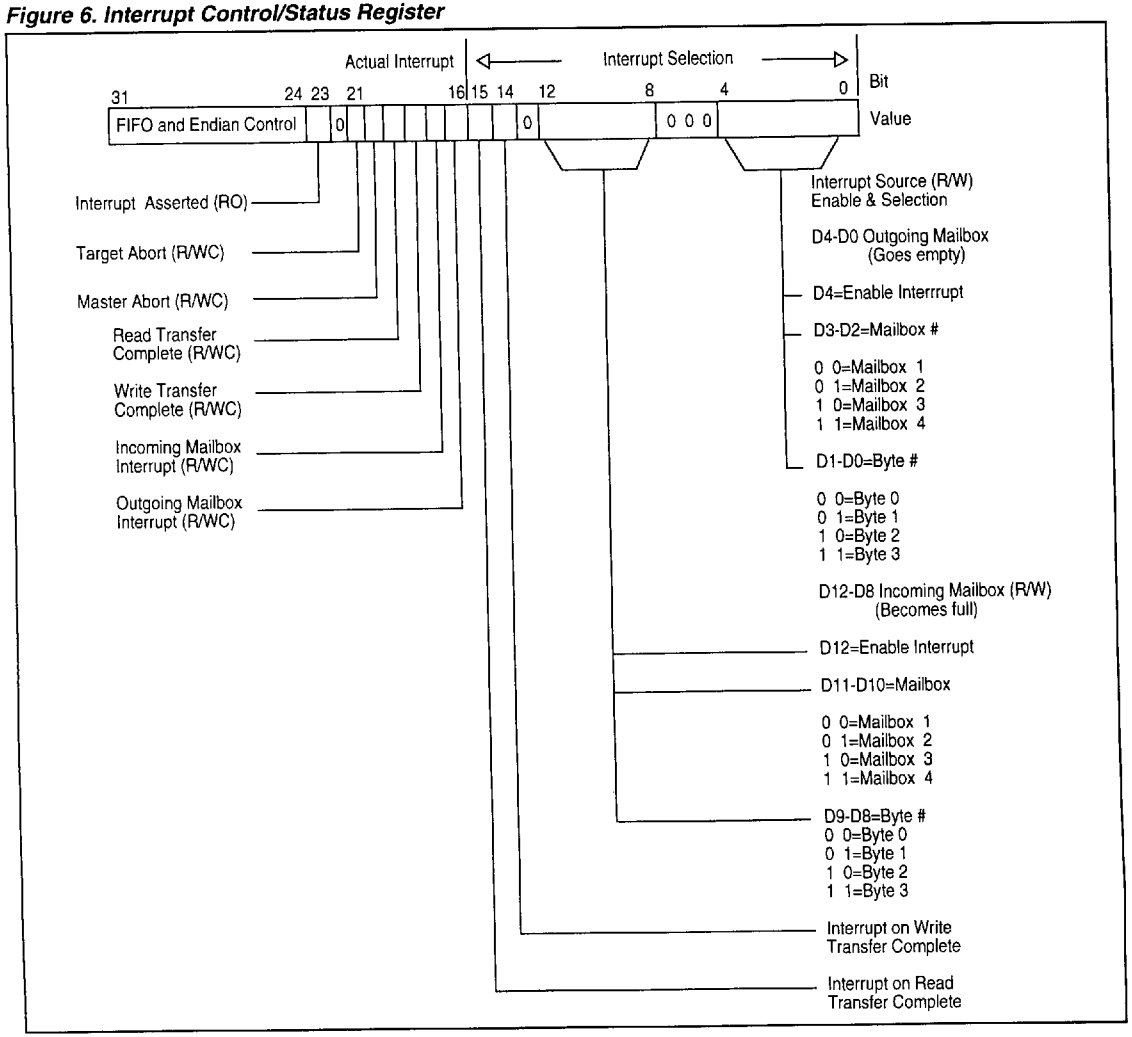
Register Name: Interrupt Control and Status
PCI Address Offset: 38h
Power-up value: 00000000h
Attribute: Read/Write (R/W), Read/Write_One_Clear (R/WC)
Size: 32 bits

This register provides the method for choosing which conditions are to produce an interrupt on the PCI bus interface, a method for viewing the cause of the interrupt, and a method for acknowledging (removing) the interrupt's assertion.

Interrupt sources:

- Write Transfer Terminal Count = zero
- Read Transfer Terminal Count = zero
- One of the Outgoing mailboxes (1,2,3 or 4) becomes empty
- One of the Incoming mailboxes (1,2,3 or 4) becomes full.
- Target Abort
- Master Abort

Figure 6. Interrupt Control/Status Register



3

Figure 7. FIFO Management and Endian Control Byte

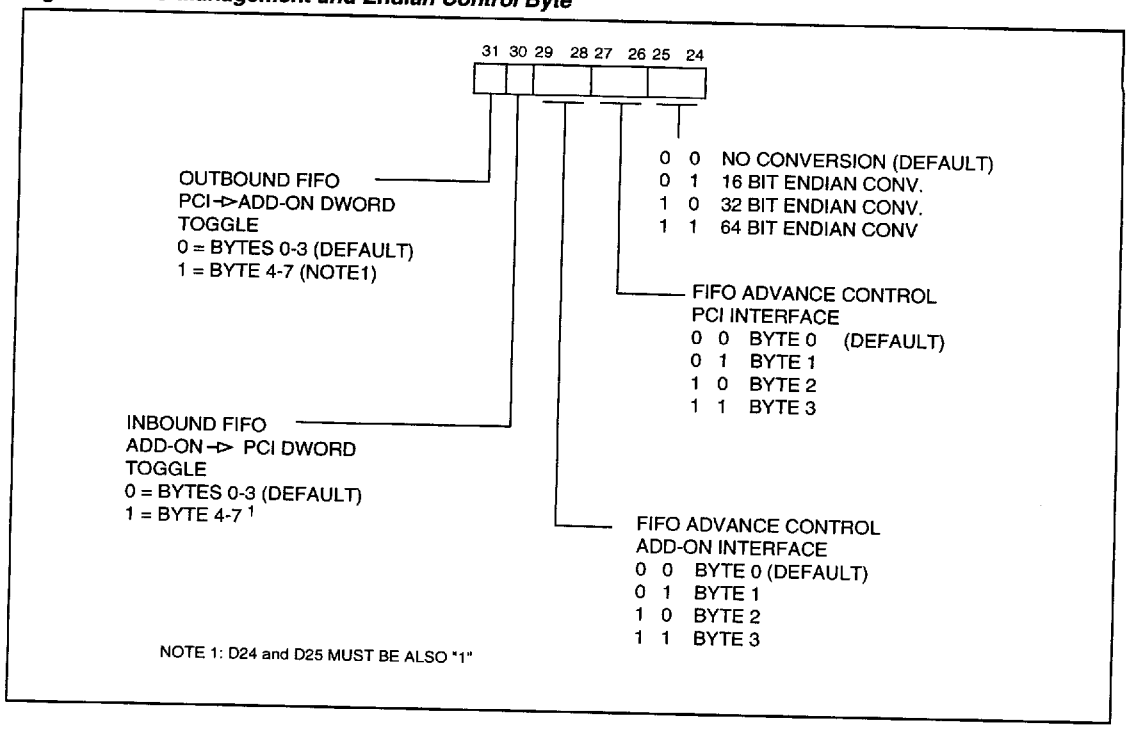


Table 3. Interrupt Control/Status Register

Bit	Description
31:24	FIFO and Endian Control.
23	Interrupt asserted. This read only status bit indicates that one or more of the four possible interrupt conditions is present. This bit is nothing more than the ORing of the interrupt conditions described by bits 19 through 16 of this register.
22	Reserved. Always zero.
21	Target Abort. This bit signifies that an interrupt has been generated due to the S5933 encountering a target abort during a PCI bus cycle while the S5933 was the current bus master. This bit operates as read or write one clear. A write to this bit with the data of "one" will cause this bit to be reset, a write to this bit with the data of "zero" will not change the state of this bit.
20	Master Abort. This bit signifies that an interrupt has been generated due to the S5933 encountering a Master Abort on the PCI bus. A master abort occurs when there is no target response to a PCI bus cycle. This bit operates as read or write one clear. A write to this bit with the data of "one" will cause this bit be reset, a write to this bit with the data of "zero" will not change the state of this bit.
19	Read Transfer Complete. This bit signifies that an interrupt has been generated due to the completion of a PCI bus master operation involving the transfer of data from the PCI bus to the Add-On. This interrupt will occur when the Master Read Transfer Count register reaches zero. This bit operates as read or write one clear. A write to this bit with the data of "one" will cause this bit to be reset; a write to this bit with the data of "zero" will not change the state of this bit.
18	Write Transfer Complete. This bit signifies that an interrupt has been generated due to the completion of a PCI bus master operation involving the transfer of data to the PCI bus from the Add-On. This interrupt will occur when the Master Write Transfer Count register reaches zero. This bit operates as read or write one clear. A write to this bit with the data of "one" will cause this bit to be reset; a write to this bit with the data of "zero" will not change the state of this bit.
17	Incoming Mailbox Interrupt. This bit is set when the mailbox selected by bits 12 through 8 of this register are written by the Add-On interface. This bit operates as read or write one clear. A write to this bit with the data of "one" will cause this bit to be reset; a write to this bit with the data as "zero" will not change the state of this bit.
16	Outgoing Mailbox Interrupt. This bit is set when the mailbox selected by bits 4 through 0 of this register is read by the Add-On interface. This bit operates as read or write one clear. A write to this bit with the data of "one" will cause this bit to be reset; a write to this bit with the data of "zero" will not change the state of this bit.
15	Interrupt on Read Transfer Complete. This bit enables the occurrence of an interrupt when the read transfer count reaches zero. This bit is read/write.
14	Interrupt on Write Transfer Complete. This bit enables the occurrence of an interrupt when the write transfer count reaches zero. This bit is read/write.
13	Reserved. Always zero.
12	Enable incoming mailbox interrupt. This bit allows a write from the incoming mailbox register identified by bits 11 through 8 to produce a PCI interface interrupt. This bit is read/write.
11:10	Incoming Mailbox Interrupt Select. This field selects which of the four incoming mailboxes is to be the source for causing an incoming mailbox interrupt. [00]b selects mailbox 1, [01]b selects mailbox 2, [10]b selects mailbox 3 and [11]b selects mailbox 4. This field is read/write.

Table 3. Interrupt Control/Status Register (Continued)

Bit	Description
9:8	Incoming Mailbox Byte Interrupt select. This field selects which byte of the mailbox selected by bits 10 and 11 above is to actually cause the interrupt. [00]b selects byte 0, [01]b selects byte 1, [10]b selects byte 2, and [11]b selects byte 3. This field is read/write.
7:5	Reserved, Always zero.
4	Enable outgoing mailbox interrupt. This bit allows a read by the Add-On of the outgoing mailbox register identified by bits 3 through 0 to produce a PCI interface interrupt. This bit is read/write.
3:2	Outgoing Mailbox Interrupt Select. This field selects which of the four outgoing mailboxes is to be the source for causing an outgoing mailbox interrupt. [00]b selects mailbox 1, [01]b selects mailbox 2, [10]b selects mailbox 3 and [11]b selects mailbox 4. This field is read/write.
1:0	Outgoing Mailbox Byte Interrupt select. This field selects which byte of the mailbox selected by bits 3 and 2 above is to actually cause the interrupt. [00]b selects byte 0, [01]b selects byte 1, [10]b selects byte 2, and [11]b selects byte 3. This field is read/write.

MASTER CONTROL/STATUS REGISTER (MCSR)

Register Name: Master Control/Status
PCI Address Offset: 3Ch
Power-up value: 000000E6h
Attribute: Read/Write, Read Only, Write Only
Size: 32 bits

This register provides for overall control of this device. It is used to enable bus mastering for both data directions as well as providing a method to perform software resets.

The following PCI bus controls are available:

- Write Priority over Read
- Read Priority over Write
- Write Transfer Enable
- Write master requests on 4 or more FIFO words available (full)
- Read transfer enable
- Read master requests on 4 or more FIFO available (empty)
- Assert reset to Add-On
- Reset Add-On to PCI FIFO flags
- Reset PCI to Add-On FIFO flags
- Reset mailbox empty full status flags
- Write external non-volatile memory

The following PCI interface status flags are provided:

- PCI to Add-On FIFO FULL
- PCI to Add-On FIFO has four or more empty locations
- PCI to Add-On FIFO EMPTY
- Add-On to PCI FIFO FULL
- Add-On to PCI FIFO has four or more words loaded
- Add-On to PCI FIFO EMPTY
- PCI to Add-On Transfer Count = Zero
- Add-On to PCI Transfer Count = Zero

3

Figure 8. Bus Master Control/Status Register

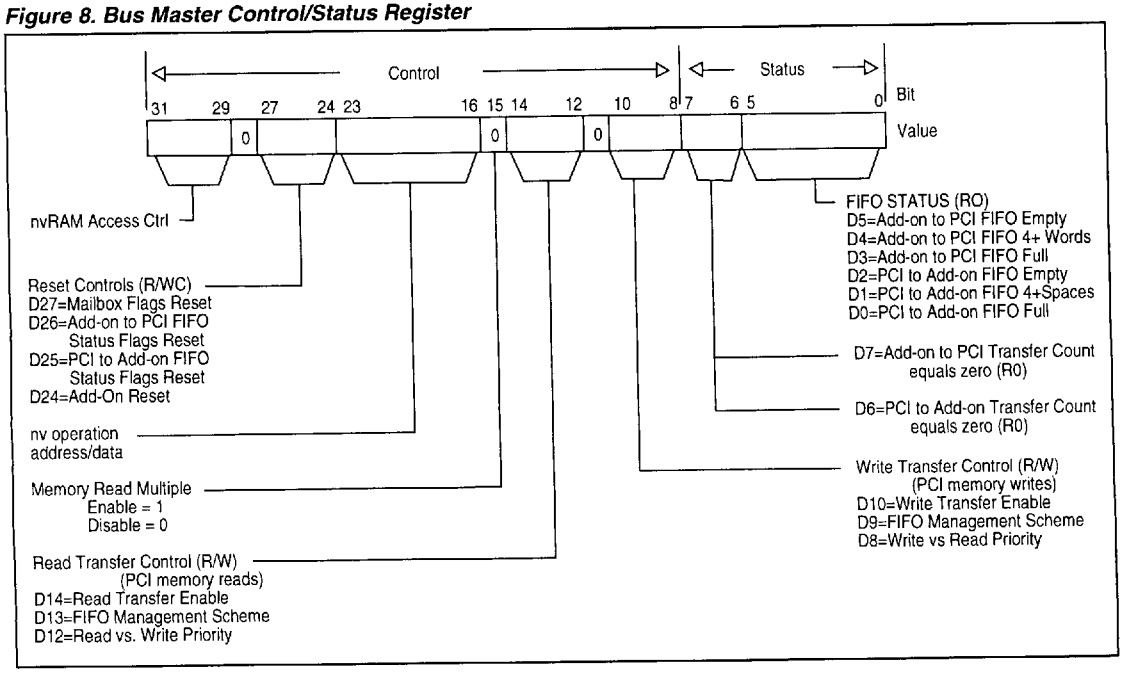


Table 4. Bus Master Control/Status Register

<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>31:29</td><td>nvRAM Access Control. This field provides a method for access to the optional external non-volatile memory. Write operations are achieved by a sequence of byte operations involving these bits and the 8-bit field of bits 23 through 16. The sequence requires that the low-order address, high order address, and then a data byte are loaded in order. Bit 31 of this field acts as a combined enable and ready for the access to the external memory. D31 must be written to a 1 before an access can begin, and subsequent accesses must wait for bit D31 to become zero (ready).
<table>
<tr><th>D31</th><th>D30</th><th>D29</th><th>W/R</th><th></th></tr>
<tr><td>0</td><td>X</td><td>X</td><td>W</td><td>Inactive</td></tr>
<tr><td>1</td><td>0</td><td>0</td><td>W</td><td>Load low address byte</td></tr>
<tr><td>1</td><td>0</td><td>1</td><td>W</td><td>Load high address byte</td></tr>
<tr><td>1</td><td>1</td><td>0</td><td>W</td><td>Begin write</td></tr>
<tr><td>1</td><td>1</td><td>1</td><td>W</td><td>Begin read</td></tr>
<tr><td>0</td><td>X</td><td>X</td><td>R</td><td>Ready</td></tr>
<tr><td>1</td><td>X</td><td>X</td><td>R</td><td>Busy</td></tr>
</table>
Cautionary note: The nonvolatile memory interface is also available for access by the Add-On interface. Accesses by both the Add-On and PCI bus to the nv memory are not directly supported by the S5933 device. Software must be designed to prevent the simultaneous access of nv memory to prevent data corruption within the memory and provide for accurate data retrieval.</td></tr>
<tr><td>28</td><td>FIFO loop back mode.</td></tr>
<tr><td>27</td><td>Mailbox Flag Reset. Writing a one to this bit causes all mailbox status flags to become reset (EMPTY). It is not necessary to write this bit as zero because it is used internally to produce a reset pulse. Since reading of this bit will always produce zeros, this bit is write only.</td></tr>
<tr><td>26</td><td>Add-On to PCI FIFO Status Reset. Writing a one to this bit causes the Add-On to PCI (Bus master memory writes) FIFO empty flag to set indicating empty and the FIFO FULL flag to reset and the FIFO Four Plus word flag to reset. It is not necessary to write this bit as zero because it is used internally to produce a reset pulse. Since reading of this bit will always produce zeros, this bit is write only.</td></tr>
<tr><td>25</td><td>PCI to Add-On FIFO Status Reset. Writing a one to this bit causes the PCI to Add-On (Bus master memory reads) FIFO empty flag to set indicating empty and the FIFO FULL flag to reset and the FIFO Four Plus words available flag to set. It is not necessary to write this bit as zero because it is used internally to produce a reset pulse. Since reading of this bit will always produce zeros, this bit is write only.</td></tr>
<tr><td>24</td><td>Add-On pin reset. Writing a one to this bit causes the reset output pin to become active. Writing a zero to this pin is necessary to remove the assertion of reset. This register bit is read/write.</td></tr>
<tr><td>23:16</td><td>Non-volatile memory address/data port. This 8-bit field is used in conjunction with bit 31, 30 and 29 of this register to access the external non-volatile memory. The contents written are either low address, high address, or data as defined by bits 30 and 29. This register will contain the external non-volatile memory data when the proper read sequence for bits 31 through 29 is performed.</td></tr>
</table>

Table 4. Bus Master Control/Status Register (Continued)

Bit	Description
15	Enable memory read multiple during S5933 bus mastering mode.
14	Read Transfer Enable. This bit must be set to a one for S5933 PCI bus master read transfers to take place. Writing a zero to this location will suspend an active transfer. An active transfer is one in which the transfer count is not zero.
13	Read FIFO management scheme. When set to a 1, this bit causes the controller to refrain from requesting the PCI bus unless it has four or more vacant FIFO locations to fill. Once the controller is granted the PCI bus or is in possession of the bus due to the write channel, this constraint is not meaningful. When this bit is zero the controller will request the PCI bus if it has at least one vacant FIFO word.
12	Read versus Write priority. This bit controls the priority of read transfers over write transfers. When set to a 1 with bit D8 as zero this indicates that read transfers always have priority over write transfers; when set to a one with D8 as one, this indicates that transfer priorities will alternate equally between read and writes.
11	Reserved. Always zero.
10	Write Transfer Enable. This bit must be set to a one for PCI bus master write transfers to take place. Writing a zero to this location will suspend an active transfer. An active transfer is one in which the transfer count is not zero.
9	Write FIFO management scheme. When set to a one this bit causes the controller to refrain from requesting the PCI bus unless it has four or more FIFO locations filled. Once the S5933 controller is granted the PCI bus or is in possession of the bus due to the write channel, this constraint is not meaningful. When this bit is zero the controller will request the PCI bus if it has at least one valid FIFO word.
8	Write versus Read priority. This bit controls the priority of write transfers over read transfers. When set to a one with bit D12 as zero this indicates that write transfers always have priority over read transfers; when set to a one with D12 as one, this indicates that transfer priorities will alternate equally between writes and reads.
7	Add-On to PCI Transfer Count Equal Zero (RO). This bit is a one to signify that the write transfer count is all zeros.
6	PCI to Add-On Transfer Count Equals Zero (RO). This bit is a one to signify that the read transfer count is all zeros.
5	Add-On to PCI FIFO Empty. This bit is a one when the Add-On to PCI bus FIFO is completely empty.
4	Add-On to PCI 4+ words. This bit is a one when there are four or more FIFO words valid within the Add-On to PCI bus FIFO.
3	Add-On to PCI FIFO Full. This bit is a one when the Add-On to PCI bus FIFO is completely full.
2	PCI to Add-On FIFO Empty. This bit is a one when the PCI bus to Add-On FIFO is completely empty.
1	PCI to Add-On FIFO 4+ spaces. This bit signifies that there are at least four empty words within the PCI to Add-On FIFO.
0	PCI to Add-On FIFO Full. This bit is a one when the PCI bus to Add-On FIFO is completely full.

PAGE(S) INTENTIONALLY BLANK

ADD-ON BUS OPERATION REGISTERS

The Add-On bus interface provides access to 18 DWORDs (72 bytes) of data, control and status information. All of these locations are accessed by asserting the Add-On bus chip select pin (SELECT#) in conjunction with either the read or write control strobes (signal pin RD# or WR#). Access to the FIFO can also be achieved through use of the dedicated pins, RDFIFO# and WRFIFO#. The dedicated pins for control of the FIFO are provided to optionally implement Direct Memory Access (DMA) on the Add-On bus, or to connect with an external FIFO.

This register group represents the primary method for communication between the Add-On and PCI buses as viewed by the Add-On. The flexibility of this arrangement allows a number of user-defined software protocols to be built. For example, data, software assigned commands, and command parameters can be exchanged between the PCI and Add-On buses using either the mailboxes or FIFOs with or without handshaking interrupts. The register structure is very similar to that of the PCI operation register set. The major difference between the PCI bus and Add-On bus register complement are the absence of bus master control registers (4) on the Add-On side and the addition of two "pass-through" registers. Table 1 lists the Add-On interface registers.

Table 1. Operation Registers — Add-On Interface

Address	Abbreviation	Register Name
00h	AIMB1	Add-On Incoming Mailbox Register #1
04h	AIMB2	Add-On Incoming Mailbox Register #2
08h	AIMB3	Add-On Incoming Mailbox Register #3
0Ch	AIMB4	Add-On Incoming Mailbox Register #4
10h	AOMB1	Add-On Outgoing Mailbox Register #1
14h	AOMB2	Add-On Outgoing Mailbox Register #2
18h	AOMB3	Add-On Outgoing Mailbox Register #3
1Ch	AOMB4	Add-On Outgoing Mailbox Register #4
20h	AFIFO	Add-On FIFO port
24h	MWAR [1]	Bus Master Write Address Register
28h	APTA	Add-On Pass-Through Address
2Ch	APTD	Add-On Pass-Through Data
30h	MRAR [1]	Bus Master Read Address Register
34h	AMBEF	Add-On Mailbox Empty/Full Status
38h	AINT	Add-On Interrupt control
3Ch	AGCSTS	Add-On General Control and Status Register
58h	MWTC [1]	Bus Master Write Transfer Count
5Ch	MRTC [1]	Bus Master Read Transfer Count

1. See Add-On Initiated Bus Mastering.

ADD-ON INCOMING MAILBOX REGISTERS (AIMBx)

Register Names: Add-On Incoming Mailboxes 1-4

Add-On Address Offset: 00h, 04h, 08h, 0Ch

Power-up value: XXXXXXXXh

Attribute: Read Only

Size: 32 bits

These four DWORD registers provide a method for receiving data, commands, or command parameters from the PCI interface. Add-On read operations to these registers may be in any width (byte, word, or DWORD). These registers are read-only. Writes to this address space have no effect. Reading from one of these registers can optionally cause a PCI bus interrupt (if desired) when the PCI interrupt control/status register is properly configured.

ADD-ON OUTGOING MAILBOX REGISTERS (AOMBx)

Register Names: Add-On Outgoing Mailboxes 1-4

Add-On Address Offset: 10h, 14h, 18h, 1Ch

Power-up value: XXXXXXXXh

Attribute: Read/Write

Size: 32 bits

These four DWORD registers provide a method for sending data, commands, or command parameters or status to the PCI interface. Add-On write operations to these registers may be in any width (byte, word, or DWORD). These registers may also be read. Writing to one of these registers can optionally cause a PCI bus interrupt (if desired) when the PCI interrupt control/status register is properly configured. Mailbox 4, byte 3 only exists as device pins on the S5933 device when used with a serial nonvolatile memory. This byte is not available if a byte-wide nv memory is used.

ADD-ON FIFO REGISTER PORT (AFIFO)

Register Name: Add-On FIFO Port

Add-On Address Offset: 20h

Power-up value: XXXXXXXXh

Attribute: Read/Write

Size: 32 bits

This location provides access to the bidirectional FIFO. Separate registers are involved when reading and writing to this location. Accordingly, it is not possible to read what was written to this location. The sequence of filling and emptying this FIFO is established by the PCI interface interrupt control and Status Register.

The FIFO's fullness may be observed by reading the master control/status register or AGCSTS register Additionally, two signal pins are provided which reveal whether data is available (RDEMPTY) or space to write into the FIFO is available (WRFULL). These signals may be used to interface with user supplied DMA logic. Caution must be exercised when using these flags for FIFO transfers involving 64 bit endian conversion since the FIFO must operate on DWORD pairs.

ADD-ON CONTROLLED BUS MASTER WRITE ADDRESS REGISTER (MWAR)

Register Name: Master Write Address

Add-On Address Offset: 24h

Power-up value: 00000000h

Attribute: Read/Write

Size: 32 bits

This register is only accessible when Add-On initiated bus mastering is enabled.

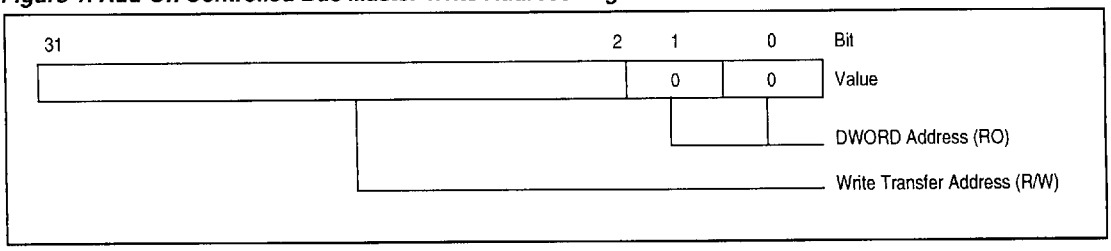
This register is used to establish the PCI address for data moving from the Add-On bus to the PCI bus during PCI bus memory write operations. It consists of a 30-bit counter with the low-order two bits hardwired as zeros. Transfers may be any non-zero byte length as defined by the transfer count register, MWTC and must begin on a DWORD boundary. This DWORD boundary starting constraint is placed upon this controller's PCI bus master transfers so that byte lane alignment can be maintained between the S5933 controller's internal FIFO data path, the Add-On interface, and the PCI bus.

Note: Applications which require a non-DWORD starting boundary will need to move the first few bytes under software program control (and without using the FIFO) to establish a DWORD boundary. After the DWORD boundary is established the S5933 can begin the task of PCI bus master data transfers.

The Master Write Address Register is continually updated during the transfer process and will always be pointing to the next unwritten location. Reading of this register during a transfer process (done when the S5933 controller is functioning as a target, i.e. not a bus master) is permitted and may be used to monitor the progress of the transfer. During the address phase for bus master write transfers, the two least significant bits presented on the PCI bus pins AD[31:0] will always be zero. This identifies to the target memory that the burst address sequence will be in a linear order rather than in an Intel 486 or Pentium™ cache line fill sequence. Also, the PCI bus address bit A1 will always be zero when this controller is the bus master. This signifies to the target that the S5933 controller is burst capable and that the target should not arbitrarily disconnect after the first data phase of this operation.

3

Figure 1. Add-On Controlled Bus Master Write Address Register



ADD-ON PASS-THRU ADDRESS REGISTER (APTA)

Register Name: Add-On Pass-Thru Address

Add-On Address Offset: 28h

Power-up value: XXXXXXXXh

Attribute: Read Only

Size: 32 bits

This register is employed when a response is desired when one of the Base address decode regions is selected during an active PCI bus cycle. When one of the base address decode registers 1-4 encounters a PCI bus cycle which selects the region defined by it, this device latches that current cycle's active address and asserts the signal PTATN# (Pass-Thru Attention). Wait states are generated on the PCI bus until either data is transferred or the PCI bus cycle is aborted by the initiator.

This register provides a method for "live" data (registered) transfers. Intended uses include the emulating of other hardware as well as enabling the connection of existing external hardware to interface to the PCI bus through the S5933.

ADD-ON PASS-THRU DATA REGISTER (APTD)

Register Name: Add-On Pass-Thru Data

Add-On Address Offset: 2Ch

Power-up value: XXXXXXXXh

Attribute: Read/Write

Size: 32 bits

This register, along with APTA described above, is employed when a response is desired should one of the Base address decode regions become selected during an active PCI bus cycle. When one of the base address decode registers 1-4 encounters a PCI bus cycle which selects the region defined by it, the APTA register will contain that current cycle's active address and the device asserts the signal PTATN# (Pass-Thru ATentioN). Wait states are generated on the PCI bus until this register is read (PCI bus writes) or this register is written (PCI bus reads).

ADD-ON CONTROLLED BUS MASTER READ ADDRESS REGISTER (MRAR)

Register Name: Master Read Address
Add-On Address Offset: 30h
Power-up value: 00000000h
Attribute: Read/Write
Size: 32 bits

This register is only accessible when Add-On initiated bus mastering is enabled.

This register is used to establish the PCI address for data moving to the Add-On bus from the PCI bus during PCI bus memory read operations. It consists of a 30-bit counter with the low-order two bits hardwired as zeros. Transfers may be any non-zero byte length as defined by the transfer count register, MRTC and must begin on a DWORD boundary. This DWORD boundary starting constraint is placed upon this controller's PCI bus master transfers so that byte lane alignment can be maintained between the S5395X controller's internal FIFO data path, the Add-On interface and the PCI bus.

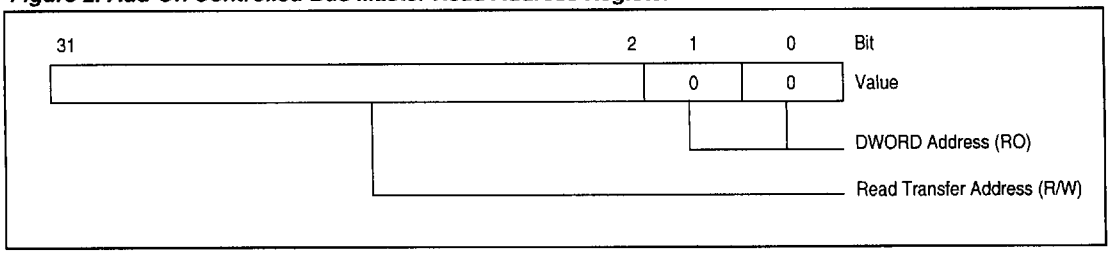
Note: Applications which require a non-DWORD starting boundary will need to move the first few bytes under software program control (and without using the FIFO) to establish a DWORD boundary. After the DWORD boundary is established the S5933 can begin the task of PCI bus master data transfers.

The Master Read Address Register is continually updated during the transfer process and will always be pointing to the next unread location. Reading of this register during a transfer process (done when the S5933 controller is functioning as a target—i.e., not a bus master) is permitted and may be used to monitor the progress of the transfer. During the address phase for bus master read transfers, the two least significant bits presented on the PCI bus AD[31:0] will always be zero. This identifies to the target memory that the burst address sequence will be in a linear order rather than in an Intel 486 or Pentium™ cache line fill sequence. Also, the PCI bus address bit A1 will always be zero when this controller is the bus master. This signifies to the target that the controller is burst capable and that the target should not arbitrarily disconnect after the first data phase of this operation.

Under certain circumstances, MRAR can be accessed from the Add-On bus instead of the PCI bus.

3

Figure 2. Add-On Controlled Bus Master Read Address Register



ADD-ON EMPTY/FULL STATUS REGISTER (AMBEF)

Register Name: Add-On Mailbox Empty/Full Status

Add-On Address Offset: 34h

Power-up value: 00000000h

Attribute: Read Only

Size: 32 bits

This register provides empty/full visibility of each byte within the mailboxes. The empty/full status for the Outgoing mailboxes are displayed on the high order 16 bits and the empty/full status for the incoming mailboxes are presented on the low order 16 bits. A value of one signifies that a given mailbox had been written by the sourcing interface but had not yet been read by the corresponding destination interface. An incoming mailbox is defined as one in which data travels from the PCI bus into the Add-On bus and an outgoing mailbox is defined as one where data goes OUT from the Add-On bus to the PCI interface.

Figure 3. Add-On Mailbox Empty/Full Status Register

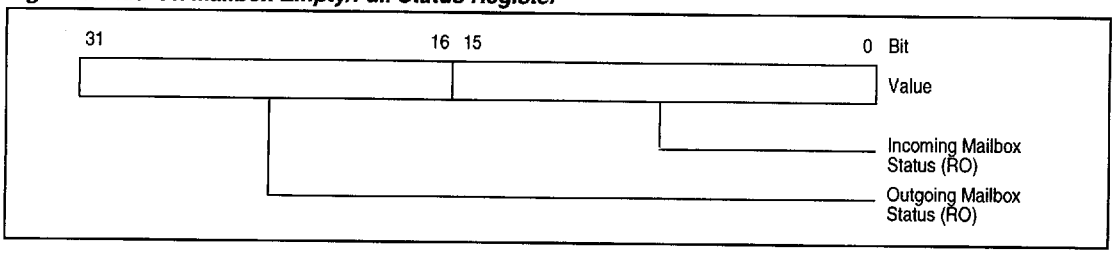


Table 2. Add-On Mailbox Empty/Full Status Register

Bit	Description
31:16	Outgoing Mailbox Status. This field indicates which outgoing mailbox registers have been written by the Add-On bus interface but have not yet been read by the PCI bus. Each bit location corresponds to a specific byte within one of the four outgoing mailboxes. A value of one for each bit signifies that the specified mailbox byte is full, a value of zero signifies empty. The mapping of these status bits to bytes within each mailbox is as follows: Bit 31 = Outgoing mailbox 4 byte 3 Bit 30 = Outgoing mailbox 4 byte 2 Bit 29 = Outgoing mailbox 4 byte 1 Bit 28 = Outgoing mailbox 4 byte 0 Bit 27 = Outgoing mailbox 3 byte 3 Bit 26 = Outgoing mailbox 3 byte 2 Bit 25 = Outgoing mailbox 3 byte 1 Bit 24 = Outgoing mailbox 3 byte 0 Bit 23 = Outgoing mailbox 2 byte 3 Bit 22 = Outgoing mailbox 2 byte 2 Bit 21 = Outgoing mailbox 2 byte 1 Bit 20 = Outgoing mailbox 2 byte 0 Bit 19 = Outgoing mailbox 1 byte 3 Bit 18 = Outgoing mailbox 1 byte 2 Bit 17 = Outgoing mailbox 1 byte 1 Bit 16 = Outgoing mailbox 1 byte 0
15:00	Incoming Mailbox Status. This field indicates which incoming mailbox registers have been written by the PCI bus but not yet been read by the Add-On interface. Each bit location corresponds to a specific byte within one of the four incoming mailboxes. A value of one for each bit signifies that the specified mailbox byte is full, a value of zero signifies empty. The mapping of these status bits to bytes within each mailbox is as follows: Bit 15 = Incoming mailbox 4 byte 3 Bit 14 = Incoming mailbox 4 byte 2 Bit 13 = Incoming mailbox 4 byte 1 Bit 12 = Incoming mailbox 4 byte 0 Bit 11 = Incoming mailbox 3 byte 3 Bit 10 = Incoming mailbox 3 byte 2 Bit 9 = Incoming mailbox 3 byte 1 Bit 8 = Incoming mailbox 3 byte 0 Bit 7 = Incoming mailbox 2 byte 3 Bit 6 = Incoming mailbox 2 byte 2 Bit 5 = Incoming mailbox 2 byte 1 Bit 4 = Incoming mailbox 2 byte 0 Bit 3 = Incoming mailbox 1 byte 3 Bit 2 = Incoming mailbox 1 byte 2 Bit 1 = Incoming mailbox 1 byte 1 Bit 0 = Incoming mailbox 1 byte 0

ADD-ON INTERRUPT CONTROL/ STATUS REGISTER (AINT)

Register Name: Add-On Interrupt Control and Status

Add-On Address Offset: 38h

Power-up value: 00000000h

Attribute: Read/Write, Read/Write_One_Clear

Size: 32 bits

This register provides the method for choosing which conditions are to produce an interrupt on the Add-On bus interface, a method for viewing the cause for the interrupt, and a method for acknowledging (removing) the interrupt's assertion.

Interrupt sources:

- One of the Incoming mailboxes (1,2,3 or 4) becomes full.
- One of the Outgoing mailboxes (1,2,3 or 4) becomes empty.
- Built-in self test issued.
- Write Transfer Count = zero
- Read Transfer Count = zero
- Target/Master Abort

Figure 4. Add-On Interrupt Control/Status Register

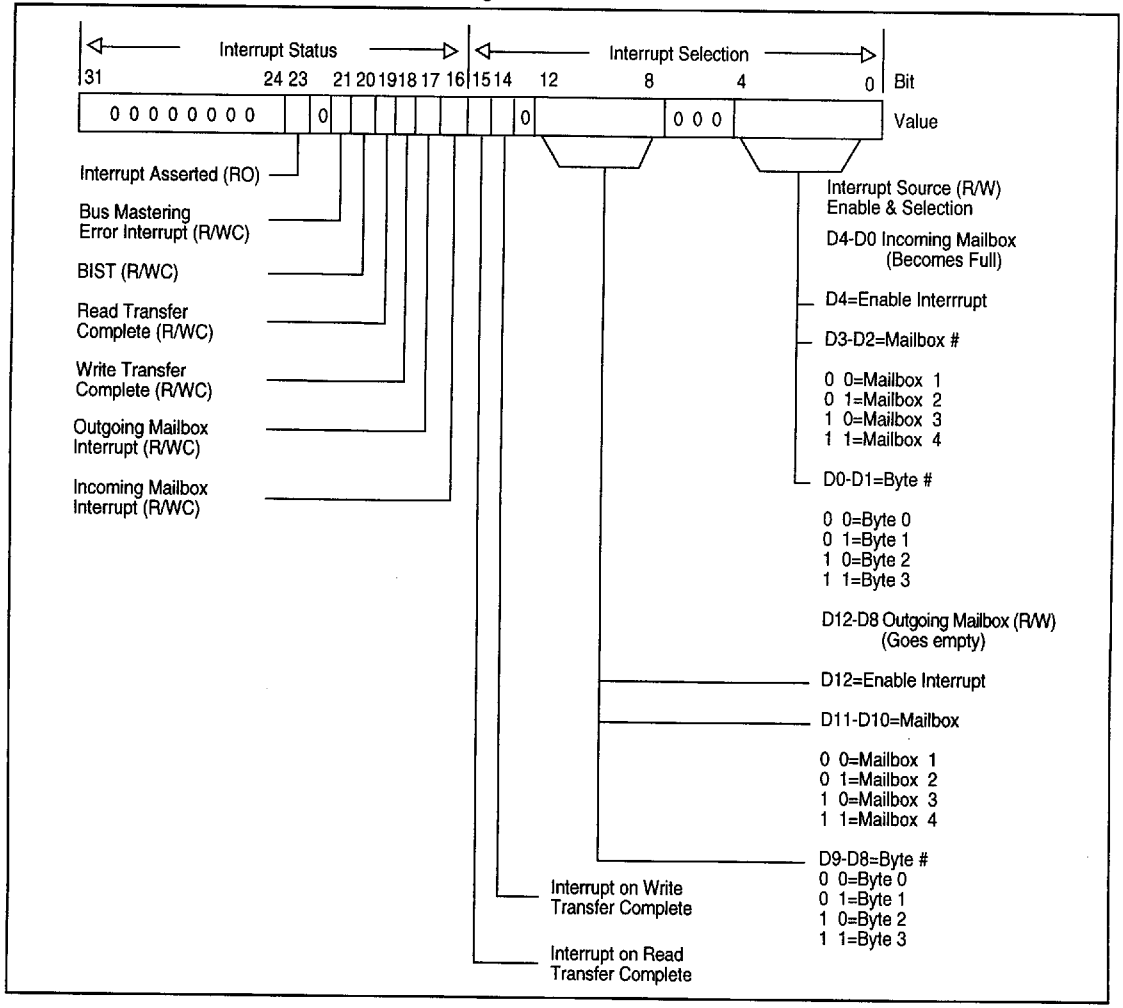


Table 3. Interrupt Control/Status Register

Bit	Description
31:24	Reserved. Always zero.
23	Interrupt asserted. This read-only status bit indicates that one or more interrupt conditions is present. This bit is nothing more than the ORing of the interrupt conditions described by bits, 20, 17 and 16 of this register.
22	Reserved. Always zero.
21	Master/Target Abort. This bit signifies that an interrupt has been generated due to the S5933 encountering a Master or Target abort during an S5933 initiated PCI bus cycle. This bit operates as read or write one clear. Writing a one to this bit causes it to be cleared. Writing a zero to this bit does nothing.
20	BIST. Built-In Self-Test interrupt. This interrupt occurs when a self test is initiated by the PCI interface writing of the BIST configuration register. This bit will stay set until cleared by writing a one to this location. Self test completion codes may be passed to the PCI BIST register by writing to the AGCSTS register.
19	Read Transfer Complete. This bit signifies that an interrupt has been generated due to the completion of a PCI bus master operation involving the transfer of data from the PCI bus to the Add-On. This interrupt will occur when the Master Read Transfer Count register reaches zero. This bit operates as read or write one clear. A write to this bit with the data of one will cause this bit to be reset; a write to this bit with the data of zero will not change the state of this bit.
18	Write Transfer Complete. This bit signifies that an interrupt has been generated due to the completion of a PCI bus master operation involving the transfer of data to the PCI bus from the Add-On. This interrupt will occur when the Master Write Transfer Count register reaches zero. This bit operates as read or write one clear. A write to this bit with the data of one will cause this bit to be reset; a write to this bit with the data of zero will not change the state of this bit.
17	Outgoing Mailbox Interrupt. This bit sets when the mailbox selected by bits 12 through 8 of this register is read by the PCI interface. This bit operates as read or write one clear. A write to this bit with the data as one will cause this bit to be reset; a write to this bit with the data as zero will not change the state of this bit.
16	Incoming Mailbox Interrupt. This bit sets when the mailbox selected by bits 5 through 0 of this register are written by the PCI interface. This bit operates as read or write one clear. A write to this bit with the data of one will cause this bit to be reset; a write to this bit with the data as zero will not change the state of this bit.
15	Interrupt on Read Transfer Complete. This bit enables the occurrence of an interrupt when the read transfer count reaches zero. This bit is read/write.
14	Interrupt on Write Transfer Complete. This bit enables the occurrence of an interrupt when the write transfer count reaches zero. This bit is read/write.
13	Reserved. Always zero.
12	Enable outgoing mailbox interrupt. This bit allows a read by the PCI of the outgoing mailbox register identified by bits 11 through 8 to produce an Add-On interface interrupt. This bit is read/write.
11:10	Outgoing Mailbox Interrupt Select. This field selects which of the four outgoing mailboxes is to be the source for causing an outgoing mailbox interrupt. [00]b selects mailbox 1, [01]b selects mailbox 2, [10]b selects mailbox 3 and [11]b selects mailbox 4. This field is read/write.
9:8	Outgoing Mailbox Byte Interrupt select. This field selects which byte of the mailbox selected by bits 11 and 10 above is to actually cause the interrupt. [00]b selects byte 0, [01]b selects byte 1, [10]b selects byte 2, and [11]b selects byte 3. This field is read/write.
7:5	Reserved. Always zero.

Table 3. Interrupt Control/Status Register (Continued)

Bit	Description
4	Enable incoming mailbox interrupt. This bit allows a write from the PCI bus to the incoming mailbox register identified by bits 3 through 0 to produce an Add-On interface interrupt. This bit is read/write.
3:2	Incoming Mailbox Interrupt Select. This field selects which of the four incoming mailboxes is to be the source for causing an incoming mailbox interrupt. [00]b selects mailbox 1, [01]b selects mailbox 2, [10]b selects mailbox 3 and [11]b selects mailbox 4. This field is read/write.
1:0	Incoming Mailbox Byte Interrupt select. This field selects which byte of the mailbox selected by bits 3 and 2 above is to actually cause the interrupt. [00]b selects byte 0, [01]b selects byte 2, and so on.

ADD-ON GENERAL CONTROL/STATUS REGISTER (AGCSTS)

Register Name: Add-On General Control and Status

Add-On Address Offset: 3Ch

Power-up value: 000000F4h (PCI initiated bus mastering)
00000034h (Add-On initiated bus mastering)

Attribute: Read/Write, Read Only, Write Only

Size: 32 bits

This register provides for overall control of the Add-On portion of this device. It is used to provide a method to perform software resets of the mailbox and FIFO flags.

The following Add-On controls are provided:

- Reset PCI to Add-On FIFO flags
- Reset Add-On to PCI FIFO flags
- Reset mailbox empty full status flags
- Write/read external non-volatile memory.

The following status flags are provided to the Add-On:

- Add-On to PCI FIFO FULL
- Add-On to PCI FIFO has four or more empty locations
- Add-On to PCI FIFO EMPTY
- PCI to Add-On FIFO FULL
- PCI to Add-On FIFO has four or more words loaded
- PCI to Add-On FIFO EMPTY

3

Figure 5. Add-On General Control/Status Register

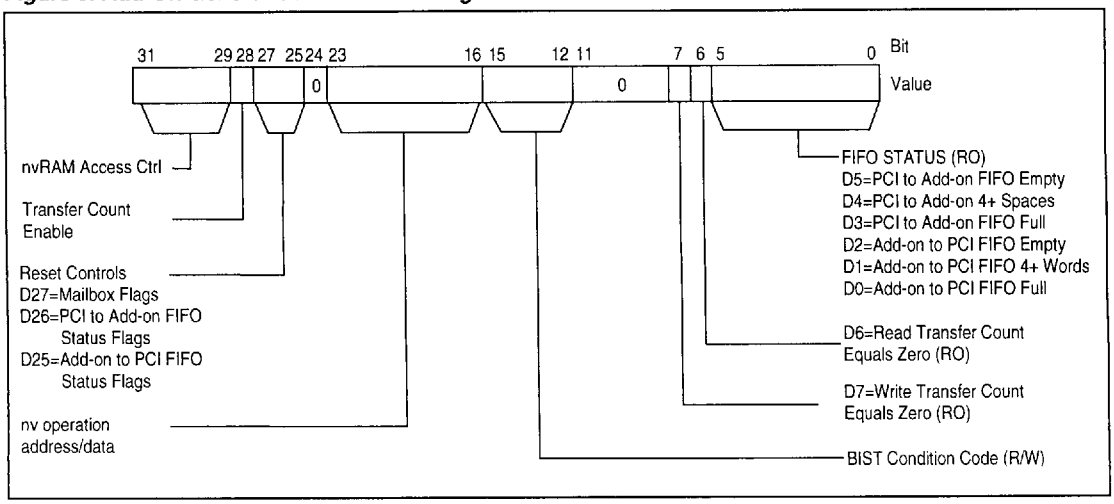


Table 4. Add-On General Control/Status Register

Bit	Description
31:29	nvRAM/EPROM Access Control. This field provides a method for access to the optional, external non-volatile memory. Write operations are achieved by a sequence of byte operations involving these bits and the 8-bit field of bits 23 through 16. The sequence requires that the low-order address, high-order address, and then a data byte be loaded in order. Bit 31 of this field acts as an enable/clock and ready for the access to the external memory. D31 must be written to a 1 before an access can begin, and subsequent accesses must wait for bit D31 to become zero (ready). D31 D30 D29 W/R 0 X X W Inactive 1 0 0 W Load low address byte 1 0 1 W Load high address byte 1 1 0 W Begin write 1 1 1 W Begin read 0 X X R Ready 1 X X R Busy Cautionary note: The non-volatile memory interface is also available for access by the PCI bus interface. Accesses by both the Add-On and PCI bus to the nv memory are not directly supported by this component. Software must be designed to prevent the simultaneous access of nv memory to prevent data corruption within the memory and provide for accurate data retrieval.
28	Transfer Count Enable. When set, transfer counts are used for Add-On initiated bus master transfers. When clear, transfer counts are ignored.
27	Mailbox Flag Reset. Writing a 1 to this bit causes all mailbox status flags to become reset (EMPTY). It is not necessary to write this bit as 0 because it is used internally to produce a reset pulse. Since reading of this bit will always produce zeros, this bit is write only.
26	PCI to Add-On FIFO Status Reset. Writing a 1 to this bit causes the Inbound (Bus master reads) FIFO empty flag to set indicating empty and the FIFO FULL flag to reset and the FIFO Four Plus spaces flag to set. It is not necessary to write this bit as 0 because it is used internally to produce a reset pulse. Since reading of this bit would always produce zeros, this bit is write only.
25	Add-On to PCI FIFO Status Reset. Writing a one to this bit causes the Outbound (Bus master writes) FIFO empty flag to set indicating empty and the FIFO FULL flag to reset and the FIFO Four Plus words available flag to reset. It is not necessary to write this bit as zero because it is used internally to produce a reset pulse. Since reading of this bit would always produce zeros, this bit is write only.
24	Reserved. Always zero.
23:16	Non-volatile memory address/data port. This 8-bit field is used in conjunction with bit 31, 30 and 29 of this register to access the external non-volatile memory. The contents written are either low address, high address, or data as defined by bits 30 and 29. This register will contain the external non-volatile memory data when the proper read sequence for bits 31 through 29 is performed.
15:12	BIST condition code. This field is directly connected to the PCI configuration self test register. Bit 15 through 12 maps with the BIST register bits 3 through 0, respectively.
11:8	Reserved. Always zero.

Table 4. Add-On General Control/Status Register (Continued)

Bit	Description
7	Add-On to PCI Transfer Count Equal Zero (RO). This bit as a one signifies that the write transfer count is all zeros. Only when Add-On initiated bus mastering is enabled.
6	PCI to Add-On Transfer Count Equals Zero (RO). This bit as a one signifies that the read transfer count is all zeros. Only when Add-On initiated bus mastering is enabled.
5	PCI to Add-On FIFO Empty. This bit is a 1 when the PCI to Add-On FIFO is empty.
4	PCI to Add-On FIFO 4+ spaces. This bit is a 1 when there are four or more open spaces in the PCI to Add-On FIFO.
3	PCI to Add-On FIFO Full. This bit is a 1 when the PCI to Add-On FIFO is full.
2	Add-On to PCI FIFO Empty. This bit is a 1 when the Add-On to PCI FIFO is empty.
1	Add-On PCI FIFO 4+ words. This bit is a 1 when there are four or more full locations in the Add-On to PCI FIFO.
0	Add-On to PCI FIFO Full. This bit is a 1 when the Add-On to PCI FIFO is full.

ADD-ON CONTROLLED BUS MASTER WRITE TRANSFER COUNT REGISTER (MWTC)

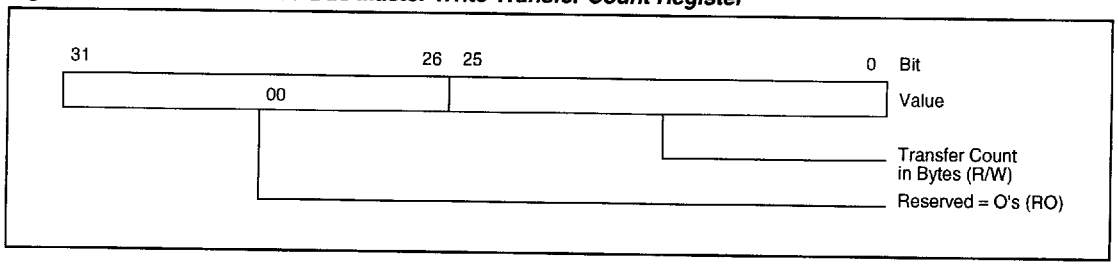
Register Name: Master Write Transfer Count
Add-On Address Offset: 58h
Power-up value: 00000000h
Attribute: Read/Write
Size: 32 bits

This register is only accessible when Add-On initiated bus mastering is enabled.

The master write transfer count register is used to convey to the S5933 controller the actual number of bytes that are to be transferred. The value in this register is decremented with each bus master PCI write operation until the transfer count reaches zero.

Upon reaching zero, the transfer operation ceases and an interrupt may be optionally generated to either the PCI or Add-On bus interface. Transfers which are not whole multiples of DWORDs in size result in a partial word ending cycle. This partial word ending cycle is possible since all bus master transfers for this controller are required to begin on a DWORD boundary.

Figure 6. Add-On Controlled Bus Master Write Transfer Count Register



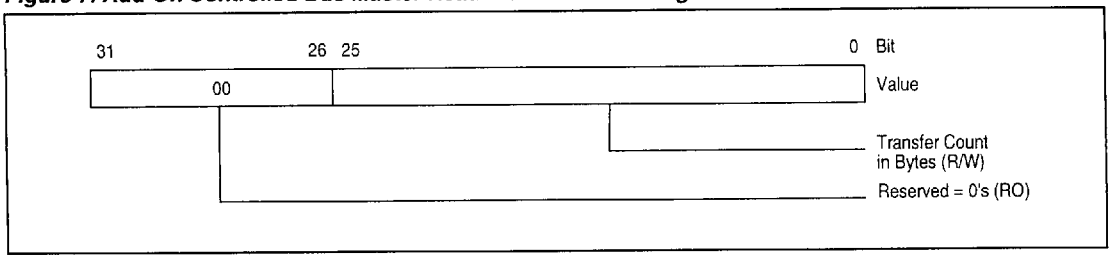
ADD-ON CONTROLLED BUS MASTER READ TRANSFER COUNT REGISTER (MRTC)

Register Name: Master Read Transfer Count
Add-On Address Offset: 5Ch
Power-up value: 00000000h
Attribute: Read/Write
Size: 32 bits

This register is only accessible when Add-On initiated bus mastering is enabled.

The master read transfer count register is used to convey to the PCI controller the actual number of bytes that are to be transferred. The value in this register is decremented with each bus master PCI read operation until the transfer count reaches zero. Upon reaching zero, the transfer operation ceases and an interrupt may be optionally generated to either the PCI or Add-On bus interface. Transfers which are not whole multiples of DWORDs in size result in a partial word ending cycle. This partial word ending cycle is possible since all bus master transfers for this controller are required to begin on a DWORD boundary.

Figure 7. Add-On Controlled Bus Master Read Transfer Count Register



PAGE(S) INTENTIONALLY BLANK

INITIALIZATION

All PCI bus agents and bridges are required to implement PCI Configuration Registers. When multiple PCI devices are present, these registers must be unique to each device in the system. The specified PCI procedure for uniquely selecting a device's configuration space involves a dedicated signal, called IDSEL, connected to each motherboard PCI bus device and PCI slot.

The host executes configuration cycles after reset to each device on the PCI bus. The configuration registers provide information on PCI agent operation and memory or I/O space requirements. These allow the PCI BIOS to enable the device and locate it within system memory or I/O space.

After a PCI reset, the S5933 can be configured for a specific application by downloading device setup information from an external non-volatile memory into the device Configuration Registers. The S5933 can also be used in a default configuration, with no external boot device.

When using a non-volatile boot memory to customize operation, 64 bytes are required for S5933 setup information. The rest of the boot device may be used to implement an Expansion BIOS, if desired. Some of the setup information is used to initialize the S5933 PCI Configuration Registers, other information is not downloaded into registers, but is used to define S5933 operation (FIFO interface, Pass-Thru operation, etc.).

PCI RESET

Immediately following the assertion of the PCI RST# signal, the Add-On reset output SYSRST# is asserted. Immediately following the deassertion of RST#, SYSRST# is deasserted. The Add-On reset output may be used to initialize state machines, reset Add-On microprocessors, or reset other Add-On logic devices.

All S5933 Operation Registers and Configuration Registers are initialized to their default states at reset. The default values for the Configuration Registers may be overwritten with the contents of an external nv boot memory during device initialization, allowing a custom device configuration. Configuration accesses by the host CPU to the S5933 produce PCI bus wait states until one of the following events occurs:

- The S5933 identifies that there is no valid boot memory (and default Configuration Register values are used).
- The S5933 finishes downloading all configuration information from a valid boot memory.

LOADING FROM BYTE-WIDE NV MEMORIES

The SNV input on the S5933 indicates what type of external boot-load device is present (if any). If SNV is tied low, a byte-wide nv memory is assumed. In this case, immediately after the PCI bus reset is deasserted, the address 0040h is presented on the nv memory interface address bus EA[15:0]. Eight PCI clocks later (240 ns at 33 MHz), data is read from the nv memory data bus EQ[7:0] and address 0041h is presented. After an additional eight PCI clocks, data is again read from EQ7:0. If both accesses read are all ones (FFh), it implies an illegal Vendor ID value, and the external nv memory is not valid or not present. In this situation, the AMCC default configuration values are used.

If either of the accesses to address 0040h and 0041h contain zeros (not FFh), the next accesses are to locations 0050h, 0051h, 0052h, and 0053h. At these locations, the data must be C0h (or C1h or C2h), FFh, E8h, and 10h, respectively, for the external nv memory to be valid. Once a valid external nv memory has been recognized, it is read, sequentially, from location 0040h to 007Fh. The appropriate data is loaded into the PCI Configuration Registers as described in Chapter 4. Some of the boot device data is not downloaded into Configuration Registers, but is used to enable features and configure S5933 operation. Upon completion of this procedure, the boot-load sequence terminates and PCI configuration accesses to the S5933 are acknowledged with the PCI Target Ready (TRDY#) output.

Table 1 lists the required nv memory contents for a valid configuration nv memory device.

Table 1. Valid External Boot Memory Contents

Address	Data	Notes
0040h-41h	not FFFFh	This is the location that the S5933 PCI Controller will load a customized vendor ID. (FFFFh is an illegal vendor ID.)
0050h	C2h, C1h or C0h	This is the least significant byte of the region which initializes the base address register #0 of the S5933 configuration register (Section 3.11). A value of C1h assigns the 16 DWORD locations of the PCI operation registers into I/O space, a value of C0h defines memory space, a value of C2h defines memory space below 1 Mbyte.
0051	FFh	Required.
0052h	E8h	Required.
0053h	10h	Required.

LOADING FROM SERIAL NV MEMORIES

SNV tied high indicates that a serial nv memory (or no external device) is present. When serial nv memories are used, data transfer is performed through a two-wire, bidirectional data transfer protocol as defined by commercial serial EEPROM/Flash offerings. These devices have the advantages of low pin counts, small package size, and economical price.

A serial nv memory is considered valid if the first serial accesses contain the correct per-byte acknowledgments (see Figure 3). If the serial per-byte acknowledgment is not observed, the S5933 determines that no external serial nv memory is present and the AMCC default Configuration Register values are used.

Two pins are used to transfer data between the S5933 PCI controller and the external serial memory: a serial clock pin, SCL, and a serial data pin, SDA. The serial clock pin is an output from the S5933, and the serial data pin is bidirectional. The serial clock is derived by dividing the PCI bus clock by 512. This means that the frequency of the serial clock is approximately 65 kHz for a 33-MHz PCI bus clock.

Communications with the serial memory involve several clock transitions. A start event signals the beginning of a transaction and is immediately followed by an address transfer. Each address/data transfer consists of 8 bits of information followed by a 1-bit acknowledgment. When the exchange is complete, a stop event is issued. Figure 1 shows the unique relationship defining both a start and stop event. Figure 2 shows the required timing for address/data with respect to the serial clock.

For random accesses, the sequence involves one clock to define the start of the sequence, eight clocks to send the slave address and read/write command, followed by a one-clock acknowledge, and so on. Figure 3 shows the sequence for a random write access requiring 29 serial clock transitions. At the clock speed for the S5933, this corresponds to one byte of data transferred approximately every 0.5 milliseconds. Read accesses may be either random or sequential. Random read access requires a dummy write to load the word address and require 39 serial clock transitions. Figure 4 shows the sequence for a random byte read.

To initialize the S5933 controller's PCI Configuration Registers, the smallest serial device necessary is a 128 x 8 organization. Although the S5933 controller only requires 64 bytes, these bytes must begin at a 64-byte address offset (0040h through 007Fh). This offset constraint permits the configuration image to be shared with a memory containing expansion BIOS code and the necessary preamble to identify an expansion BIOS. The largest serial device which may be used is 2 Kbytes.

Figure 1. Serial Interface Definition of Start and Stop

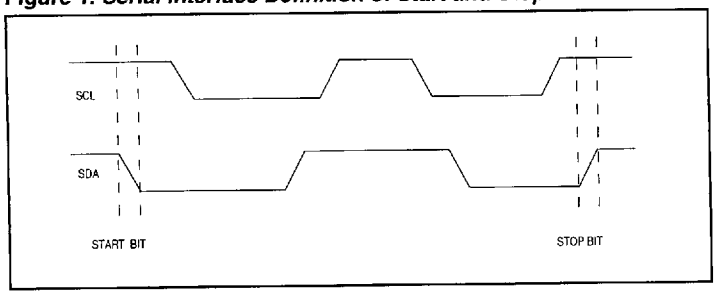


Figure 2. Serial Interface Clock/Data Relationship

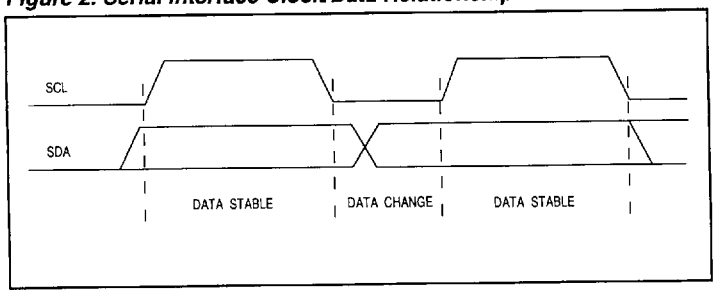


Figure 3. Serial Interface Byte Access — Write

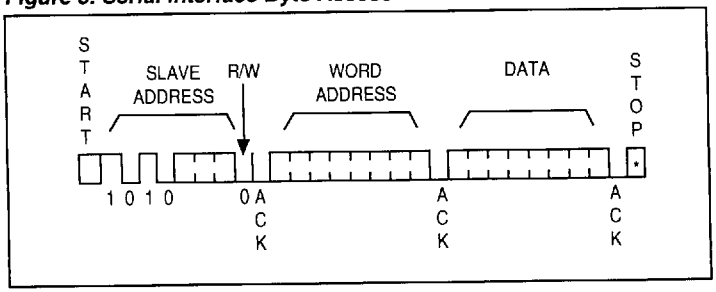
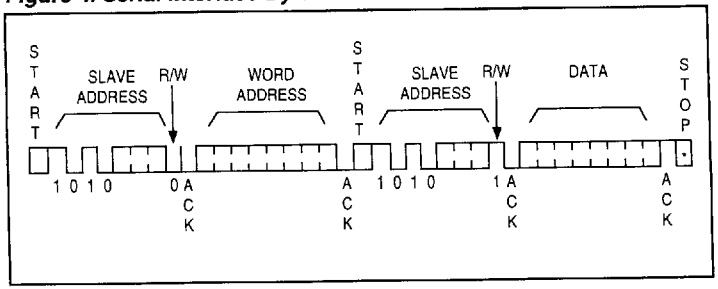


Figure 4. Serial Interface Byte Access — Read



PCI BUS CONFIGURATION CYCLES

Cycles beginning with the assertion IDSEL and FRAME# along with the two configuration command states for C/BE[3:0] (configuration read or write) access an individual device's configuration space. During the address phase of the configuration cycle just described, the values of AD0 and AD1 identify if the access is a Type 0 configuration cycle or a Type 1 configuration cycle. Type 0 cycles have AD0 and AD1 equal to 0 and are used to access PCI bus agents. Type 1 configuration cycles are intended only for bridge devices and have AD0 as a 1 with AD1 as a 0 during the address phase.

The S5933 PCI device is a bus agent (not a bridge) and responds only to a Type 0 configuration accesses. Figure 5 depicts the state of the AD bus during the address phase of a Type 0 configuration access. The S5933 controller does not support the multiple function numbers field (AD[10:8]) and only responds to the all-zero function number value.

The configuration registers for the S5933 PCI controller can only be accessed under the following conditions:

- IDSEL high (PCI slot unique signal which identifies access to configuration registers) along with FRAME# low.
- Address bits A0 and A1 are 0 (Identifies a Type 0 configuration access).
- Address bits A31-A11 are ignored.
- Address bits A8, A9, and A10 are 0 (Function number field of zero supported).
- Command bits, C/BE[3:0]# must identify a configuration cycle command (101X).

Figure 6 describes the signal timing relationships for configuration read cycles. Figure 7 describes configuration write cycles.

Figure 5. PCI AD Bus Definition During a Type 0 Configuration Access

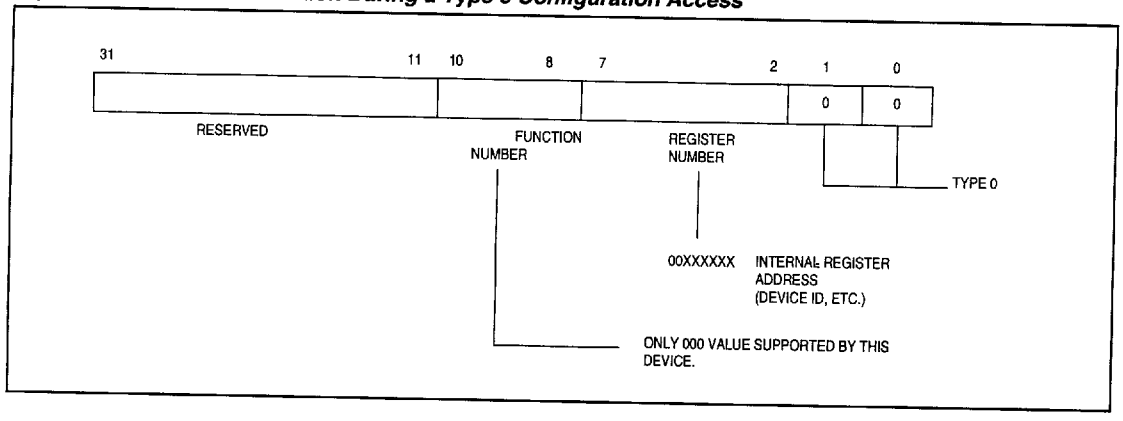


Figure 6. Type 0 Configuration Read Cycles

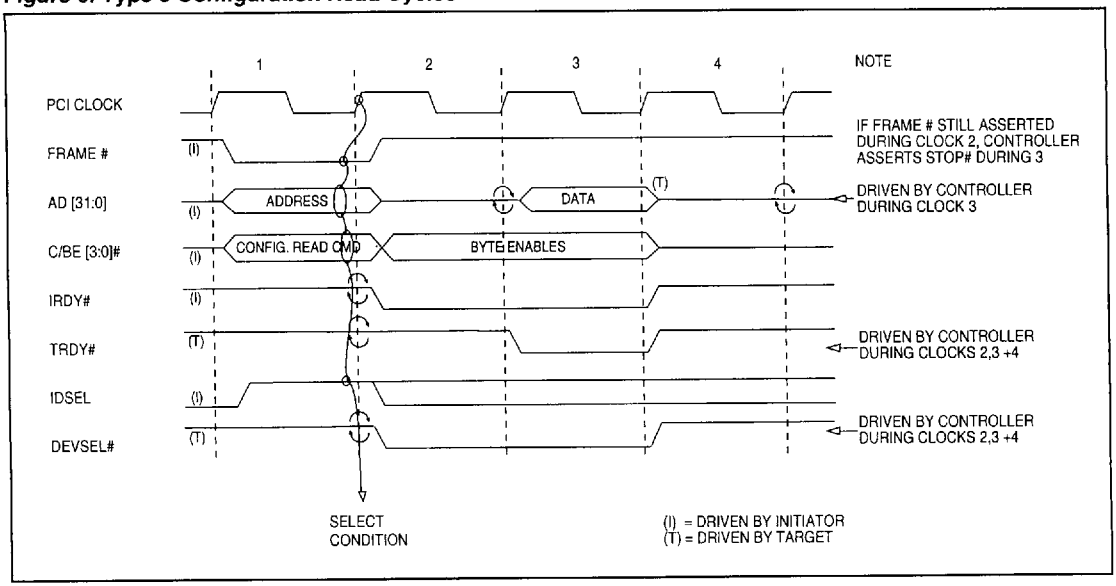
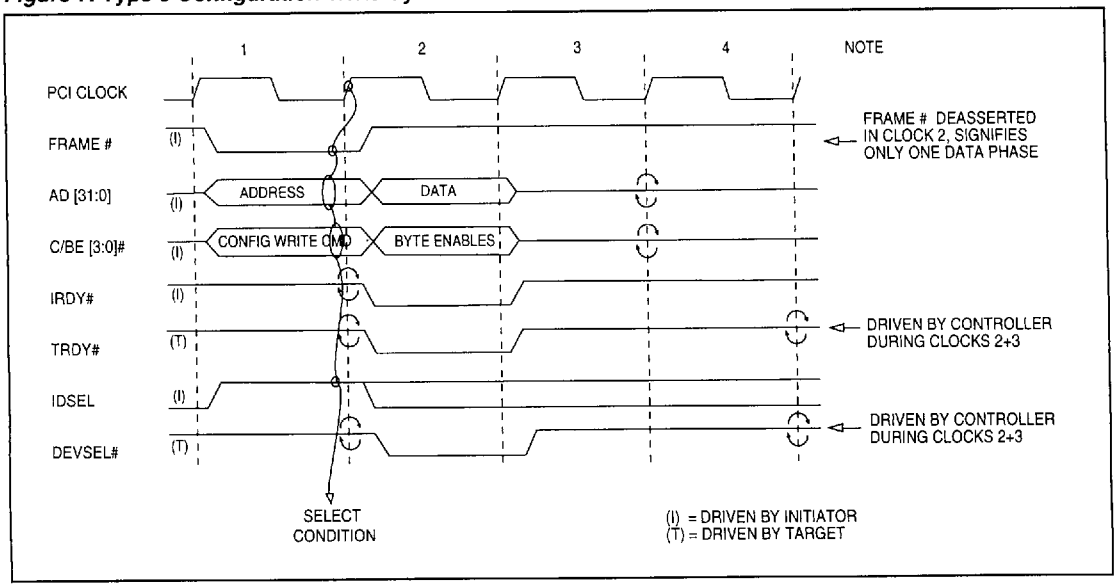


Figure 7. Type 0 Configuration Write Cycles



EXPANSION BIOS ROMS

This section provides an example of a typical PC-compatible expansion BIOS ROM. Address offsets 0040h through 007Fh represent the portion of the external nv memory used to boot-load the S5933 controller. Whether the expansion ROM is intended to be executable code is determined by the contents of the first three locations (starting at offset 0h) and a byte checksum over the defined length. The defined length is specified in the byte at address offset 0002h. Table 2 lists each field location by its address offset, its length, its value, and description.

Table 2. PC Compatible Expansion ROM

Byte Offset	Byte Length	Binary Value	Description	Example
0h	1	55h	BIOS ROM signature byte 1	55h
1h	1	AAh	BIOS ROM signature byte 2	AAh
2h	1	var.	Length in multiples of 512 bytes	01h
3h	4	var.	Entry point for INIT function.	
7h-17h	17h	var.	Reserved (application unique data)	
18h-19h	2	var.	Pointer to PCI Data Structure (see Table 3)	
20h-3Fh	32h	var	user-defined	
The following represents the boot-load image for the S5933 controller's PCI configuration register :				
40h	2	[your vendor ID]		10e8h
42h	2	[your device ID]		4750h
44h	1	not used		
45h	1	[Bus Master Config.]		80h
46h	2	not used		
48h	1	[your revision ID]		00h
49h	3	[your class code]		FF0000h
4Ch	1	not used		
4Dh	1	[your latency timer #]		00h
4Eh	1	[your header type]		00h
4Fh	1	[self-test if desired]		80h or 00h
50h	1	C0h, C1h or C2h		C0h, C1h or C2h
51h	1	FFh		FFh
52h	1	E8h		E8h
53h	1	10h		10h
54h	4	[base addr. #1]		xxxxxxxxh
58h	4	[base addr. #2]		xxxxxxxxh
5Ch	4	[base addr. #3]		xxxxxxxxh
60h	4	[base addr. #4]		xxxxxxxxh
64h	4	[base addr. #5]		xxxxxxxxh

Table 2. PC Compatible Expansion ROM (Continued)

Byte Offset	Byte Length	Binary Value	Description	Example
68h	8	not used		
70h	4	[Expansion ROM base addr.]	(example shows 32K bytes)	FFFF8001h
74h	8	not used		
7Ch	1	[Interrupt line]		0Ch
7Dh	1	[Interrupt pin]		01h
7Eh	1	[Min-Grant]		00h
7Fh	1	[Max_lat]		00h
80h — (1FFh), or (2FFh), or (3FFh), etc.	application specific Byte checksum, location dependent on value for length field at offset 0002h.			

A 16-bit pointer at location 18h of the PC expansion ROM identifies the start offset of the PCI data structure. The PCI data structure is shown in Table 3 and contains various vendor, product, and program evolutions. If a valid external nv memory is identified by the S5933, the PCI data structure is used to configure the S5933. The PCI data structure is not necessary for this device to operate. If no external nv memory is implemented, the S5933 boots with the default configuration values.

Note: If a serial BIOS ROM is used, the access time for large serial devices should be considered, since it may cause a lengthy system delay during initialization. For example, a 2-Kbyte serial device takes about 1 second to be read. Many systems, even when BIOS ROMs are ultimately shadowed into system RAM, may read this memory space twice (once to validate its size and checksum, and once to move it into RAM). Execution directly from a serial BIOS ROM, although possible, may be unacceptably slow.

Table 3. PCI Data Structure

Byte Offset	Byte Length	Binary Value	Description
0h	4	'PCIR'	Signature, the ASCII string 'PCIR' where 'P' is at offset 0, 'C' at offset 1, and so on.
4h	2	var.	Vendor Identification
6h	2	var.	Device Identification
8h	2	var.	Pointer to Vital Product Data
Ah	2	var.	PCI Data Structure Length (starts with signature field)
Ch	1	var.	PCI Data Structure Revision (=0 for this definition)
Dh	3	var.	Class Code
10h	2	var.	Image Length
12h	2	var.	Revision Level
14h	1	var.	Code Type
15h	1	var.	Indicator (bit D7=1 signifies "last image")
16h	2	0000h	Reserved

PAGE(S) INTENTIONALLY BLANK

PCI BUS INTERFACE

This section describes the various events which occur on the S5933 PCI bus interface. Since the S5933 controller functions as both a target (slave) and an initiator (master), signal timing detail is given for both situations this Section presents the signal relationships involved in performing basic read or write transfers on the PCI bus and also describes the different ways these cycles may complete.

PCI BUS TRANSACTIONS

Because the PCI bus has multiplexed address/data pins, AD[31:0], each PCI bus transaction consists of two phases: Address and Data. An address phase is defined by the clock period when the signal FRAME# transitions from inactive (high) to active (low). During the address phase, a bus command is also driven by the initiator on signal pins C/BE[3:0]#. If the command indicates a PCI read, the clock cycle following the address phase is used to perform a "bus turn-around" cycle. A turn-around cycle is a clock period in which the AD bus is not driven by the initiator or the target device. This is used to avoid PCI bus contention. For a write command, a turn-around cycle is not needed, and the bus goes directly from the address phase to the data phase.

All PCI bus transactions consist of an address phase (described above), followed by one or more data phases. The address phase is only one PCI clock long and the bus cycle information (address and command) is latched internally by the S5933. The number of data phases depends on how many data transfers are desired or are possible with a given initiator-target pair. A data phase consists of at least one PCI clock. FRAME# is deasserted to indicate that the final data phase of a PCI cycle is occurring. Wait states may be added to any data phase (each wait state is one PCI clock).

The PCI bus command presented on the C/BE[3:0]# pins during the address phase can represent 16 possible states. Table 1 lists the PCI commands and identifies those which are supported by the S5933 controller as a target and those which may be produced by the S5933 controller as an initiator. A "Yes" in the "Supported As Target" column in Table 1 indicates the S5933 controller asserts the signal DEVSEL# when that command is issued along with the appropriate PCI address. Two commands are supported by the S5933 controller as an initiator: Memory Read and Memory Write.

The completion or termination of a PCI cycle can be signaled in several ways. In most cases, the completion of the final data phase is indicated by the assertion of ready signals from both the target (TRDY#) and initiator (IRDY#) while FRAME# is inactive. In some cases, the target is not be able to continue or support a burst transfer and asserts the STOP# signal. This is referred to as a target disconnect. There are also cases where an addressed device does not exist, and the signal DEVSEL# never becomes active. When no DEVSEL# is asserted in response to a PCI cycle, the initiator is responsible for ending the cycle. This is referred to as a master abort. The bus is returned to the idle phase when both FRAME# and IRDY# are deasserted.

Table 1. Supported PCI Bus Commands

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

1. Memory Read Multiple and Read Line are treated as Memory Reads.
2. Memory Write & Invalidate commands are treated as Memory Writes.
3. Must be enabled by bit 15 MCSR.

PCI Burst Transfers

The PCI bus, by default, expects burst transfers to be executed. To successfully perform a burst transfer, both the initiator and target must order their burst address sequence in an identical fashion. There are two different ordering schemes: linear address incrementing and 80486 cache line fill sequencing. The exact ordering scheme for a bus transaction is defined by the state of the two least significant AD lines during the address phase. The decoding for these lines is shown below:

AD[1:0]	Burst Order
0 0	Linear sequence
0 1	Reserved
1 0	Cacheline Wrap Mode
1 1	Reserved

The S5933 supports both the linear and the cache line burst ordering. When the S5933 controller is an initiator, it always employs a linear ordering.

Some accesses to the S5933 controller (as a target) can not be burst transfers. For example, the S5933 does not allow burst transfers when accesses are made to the configuration or operation registers (including the FIFO as a target). Attempts to perform burst transfers to these regions cause STOP# to be asserted during the first data phase. The S5933 completes the initial data phase successfully, but asserting STOP# indicates that the next access needs to be a completely new cycle. Accesses to memory or I/O regions defined by the Base Address Registers 1-4 may be bursts, if desired.

PCI Read Transfers

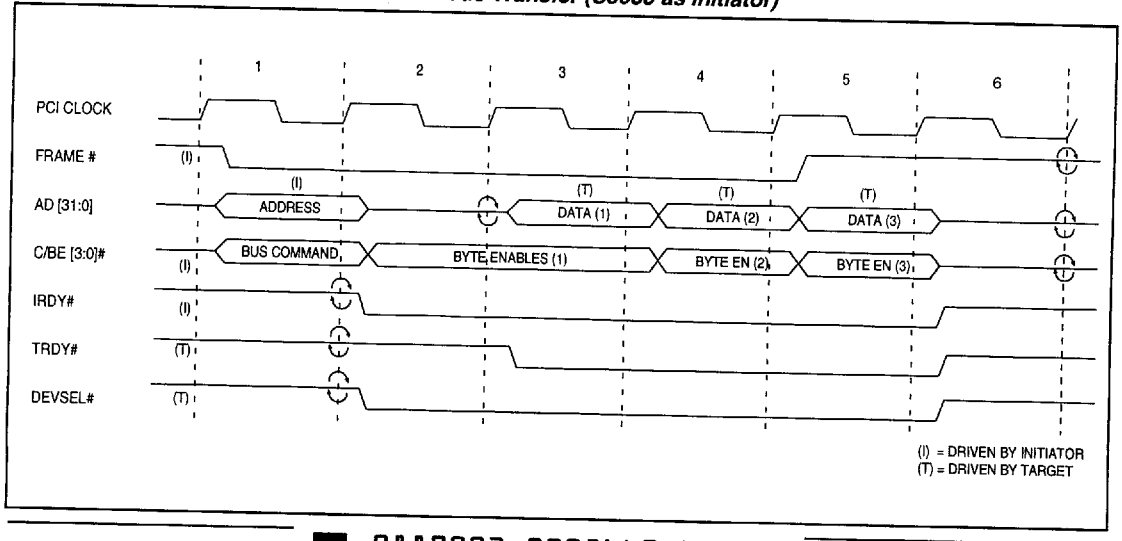
The S5933 responds to PCI bus memory or I/O read transfers when it is selected (target). As a PCI bus initiator, the S5933 controller may also produce PCI bus memory read operations.

Figure 1 depicts the fastest burst read transfer possible for the PCI bus. The timings shown in Figure 1 are representative of the S5933 as a PCI initiator with a fast, zero-wait-state memory target. The signals driven by the S5933 during the transfer are FRAME#, C/BE[3:0]#, and IRDY#. The signals driven by the target are DEVSEL# and TRDY#. AD[31:0] are driven by both the target and initiator during read transactions (only one during any given clock). Clock period 2 is a required bus turn-around clock which ensures bus contention between the initiator and target does not occur.

Targets drive DEVSEL# and TRDY# after the end of the address phase (boundary of clock periods 1 and 2 of Figure 1). TRDY# is not driven until the target can provide valid data for the PCI read. When the S5933 becomes the PCI initiator, it attempts to perform sustained zero-wait state burst reads until one of the following occurs:

- The memory target aborts the transfer
- PCI bus grant (GNT#) is removed
- The PCI to Add-On FIFO becomes full
- A higher priority (Add-On to PCI) S5933 transfer is pending (if programmed for priority)
- The read transfer byte count reaches zero
- Bus mastering is disabled from the Add-On interface

Figure 1. Zero Wait State Burst Read PCI Bus Transfer (S5933 as Initiator)



Read accesses from the S5933 operation registers (S5933 as a target) are shown in Figure 2. The S5933 conditionally asserts STOP# in clock period 3 if the initiator keeps FRAME# asserted during clock period 2 with IRDY# asserted (indicating a burst is being attempted). Wait states may be added by the initiator by not asserting the signal IRDY# during clock 3 and beyond. If FRAME# remains asserted, but IRDY# is not asserted, the initiator is just adding wait states, not necessarily attempting a burst.

There is only one condition where accesses to S5933 operation registers do not return TRDY# but do assert STOP#. This is called a target-initiated termination or target disconnect and occurs when a read attempt is made to an empty S5933 FIFO. The assertion of STOP# without the assertion of TRDY# indicates that the initiator should retry the operation later.

When burst read transfers are attempted to the S5933 operation registers, STOP# is asserted during the first data transfer to indicate to the initiator that no further transfers (data phases) are possible. This is a target-initiated termination where the target disconnects after the first data transfer. Figure 3 shows the signal relationships during a burst read attempt to the S5933 operation registers.

Figure 2. Single Data Phase PCI Bus Read of S5933 Registers (S5933 as Target)

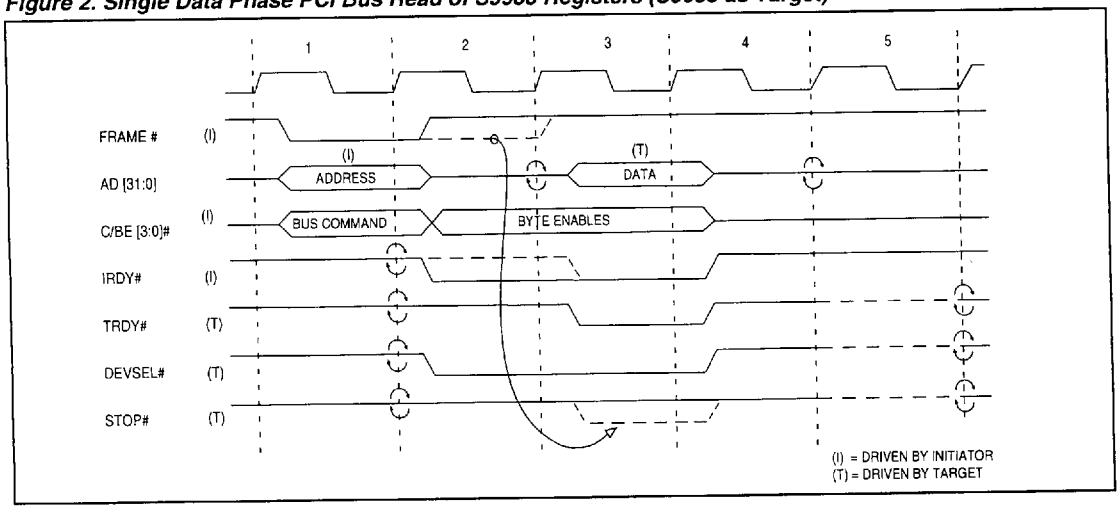
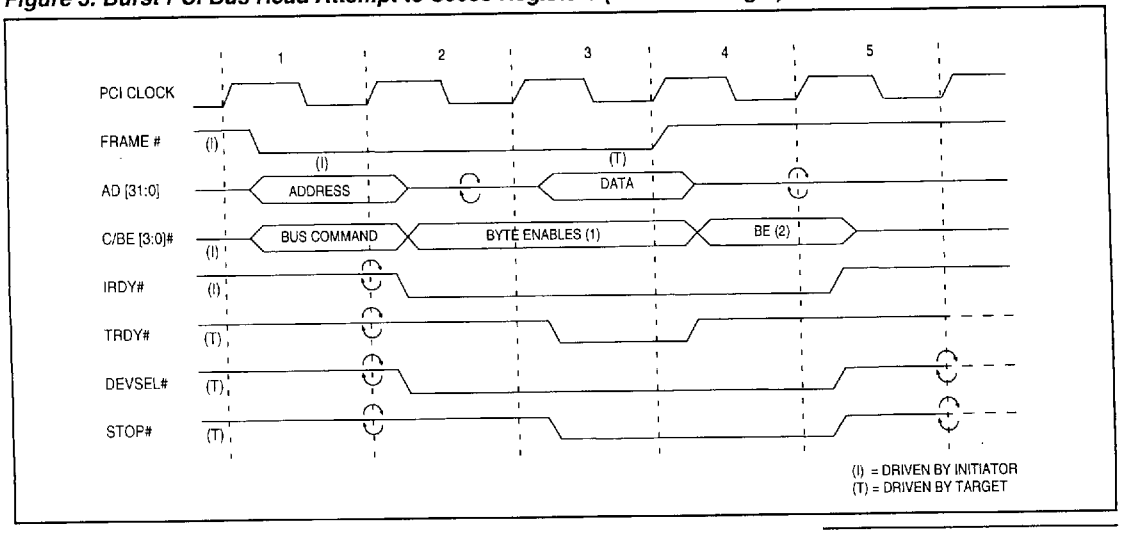


Figure 3. Burst PCI Bus Read Attempt to S5933 Registers (S5933 as Target)



PCI Write Transfers

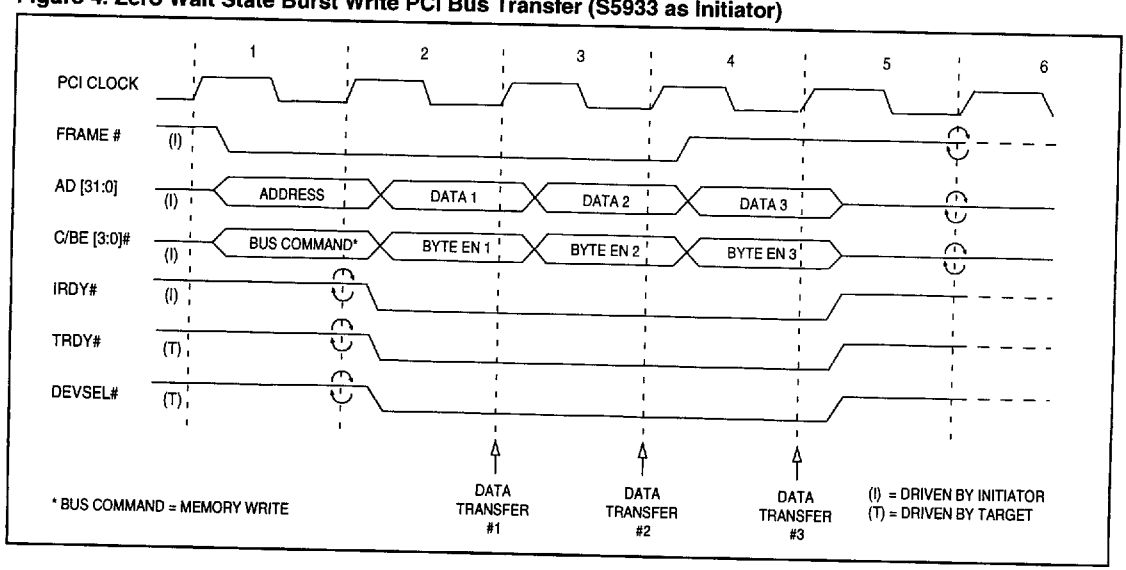
Write transfers on the PCI bus are one clock period shorter than read transfers. This is because the AD[31:0] bus does not require a turn-around cycle between the address and data phases. When the S5933 is accessed (target), it responds to a PCI bus memory or I/O transfers. As a PCI initiator, the S5933 controller can also execute PCI memory write operations.

The timing diagram in Figure 4 represents an S5933 initiator PCI write operation transferring to a fast, zero-wait-state memory target. The signals driven by the S5933 during the transfer are FRAME#, AD[31:0], C/BE[3:0]#, and IRDY#. The signals driven by the target are DEVSEL# and TRDY#. As with PCI reads, targets assert DEVSEL# and TRDY# after the clock defining the end of the address phase (boundary of clock periods 1 and 2 of Figure 4). TRDY# is not driven until the target has accepted the data for the PCI write. When the S5933 becomes the PCI initiator, it attempts sustained zero-wait state burst writes until one of the following occurs:

- The memory target aborts the transfer
- PCI bus grant (GNT# is removed)
- The Add-On to PCI FIFO becomes empty
- A higher priority (PCI to Add-On) S5933 transfer is pending (if programmed for priority)
- The write transfer byte count reaches zero
- Bus mastering is disabled from the Add-On interface

Write accesses to the S5933 operation registers (S5933 as a target) are shown in Figure 5. Here, the S5933 asserts the signal STOP# in clock period 3. STOP# is asserted because the S5933 supports fast, zero-wait-state write cycles but does not support burst writes to operation registers. Wait states may be added by the initiator by not asserting the signal IRDY# during clock 2 and beyond. There is only one condition where writes to S5933 operation registers do not return TRDY# (but do assert STOP#). This is called a target-initiated termination or target disconnect and occurs when a write attempt is made to a full S5933 FIFO. As with the read transfers, the assertion of STOP# without the assertion of TRDY# indicates the initiator should retry the operation later.

Figure 4. Zero Wait State Burst Write PCI Bus Transfer (S5933 as Initiator)



Master-Initiated Termination

Occasionally, a PCI transfer must be terminated by the initiator. Typically, the initiator terminates a transfer upon the successful completion of the transfer. Sometimes, the initiator's bus mastership is relinquished by the bus arbiter (GNT# is removed), often because another device requires bus ownership. This is called initiator preemption and is discussed in later Sections. When the S5933 is an initiator and does not observe a DEVSEL# response to its assertion of FRAME#, it terminates the cycle (master abort).

Normal Cycle Completion

A successful data transfer occurs when both the initiator and target assert their respective ready signals, IRDY# and TRDY#. The last data phase is indicated by the initiator when FRAME# is deasserted during a data transfer. A normal cycle completion occurred if the target does not assert STOP#. Figure 6 shows the signal relationships defining a normal transfer completion.

Figure 5. Single Data Phase PCI Bus Write of S5933 Registers (S5933 as Target)

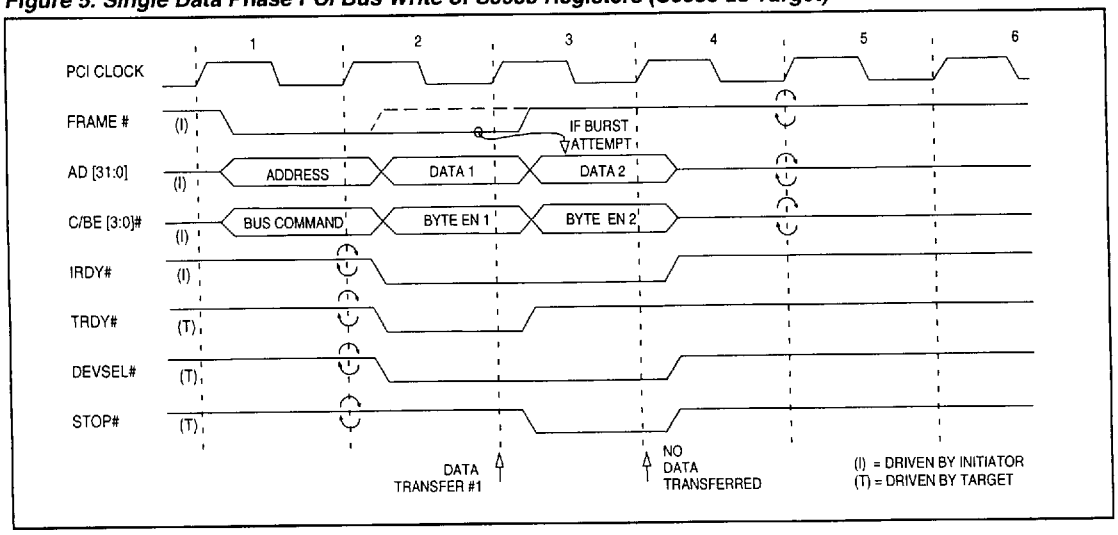
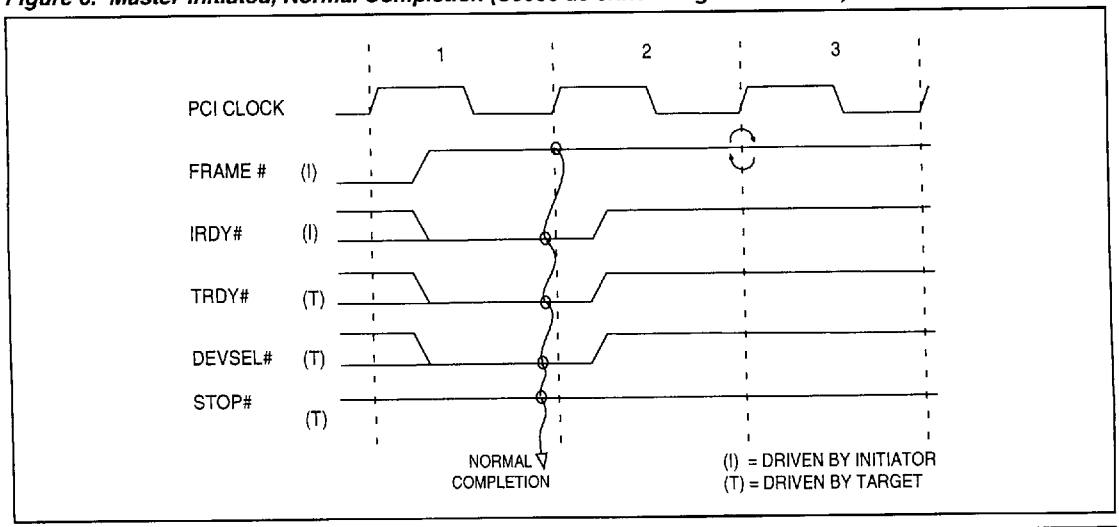


Figure 6. Master-Initiated, Normal Completion (S5933 as either Target or Initiator)



Initiator Preemption

A PCI initiator (bus master) is said to be preempted when the system platform deasserts the initiator's bus grant signal, GNT#, while it still requests the bus (REQ# asserted). This situation occurs if the initiator's latency timer expires and the system platform (bus arbitrator) has a bus master request from another device. The S5933 Master Latency Timer register controls the S5933 responsiveness to the removal of a bus grant (preemption). The presence of a Master Latency Timer register can cause two preemption situations:

1) Removal of GNT# when the latency timer is non-zero (S5933 is guaranteed to still "own the bus").

2) Removal of the GNT# after the latency timer has expired.

The first situation is depicted in Figure 7, when the latency timer has not expired. Preemption with a zero or expired latency timer is shown in Figure 8.

Figure 7. Master Initiated Termination Due to Preemption and Latency Timer Active (S5933 as Master)

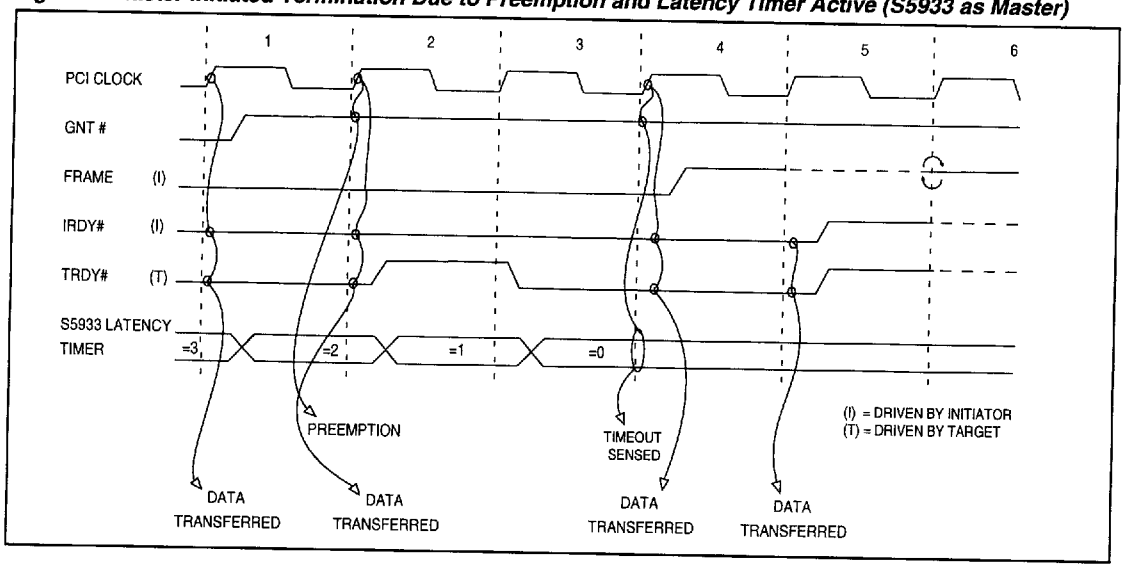
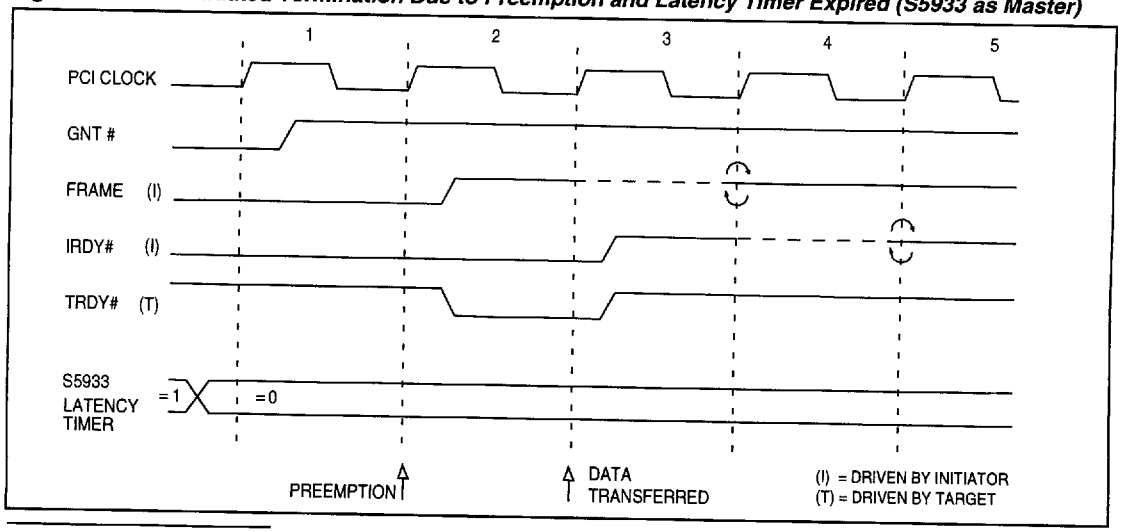


Figure 8. Master Initiated Termination Due to Preemption and Latency Timer Expired (S5933 as Master)



Master Abort

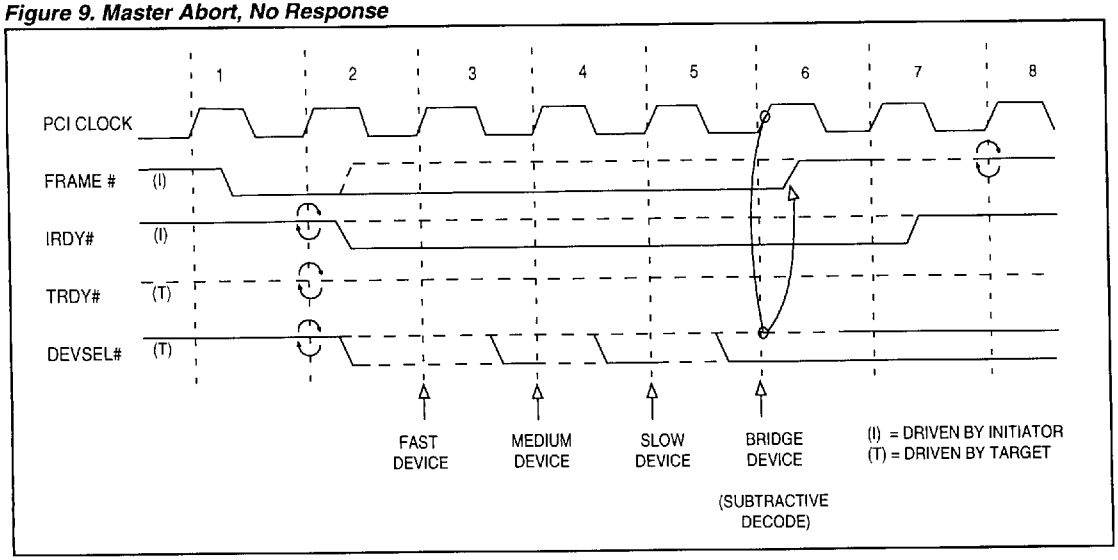
PCI accesses to nonexistent or disabled targets never observe DEVSEL# being asserted. In this situation, it is necessary for the initiator to abort the transaction (master abort). As an initiator, S5933 waits for six clock periods after asserting FRAME# to determine whether a master abort is required. These six clock periods allow slow targets, which may require several bus clocks before being able to assert DEVSEL#, to respond. It is also possible a PCI bridge device is present which employs "subtractive" decoding. A device which does a subtractive decode asserts DEVSEL#, claiming the cycle, when it sees that no other device has asserted it three clocks after the address phase.

If DEVSEL# is not asserted, the S5933 deasserts FRAME# (if asserted) upon the sixth clock period (Figure 9). IRDY# is deasserted by the S5933 during the next clock. The occurrence of a master abort causes the S5933 to set bit 13 (Master Abort) of the PCI Status Register, indicating an error condition.

Target-Initiated Termination

There are situations where the target may end a transfer prematurely. This is called "target-initiated termination." Target terminations fall into three categories: disconnect, retry, and target abort. Only the disconnect termination completes a data transfer.

Figure 9. Master Abort, No Response



3

Target Disconnects

There are many situations where a target may disconnect. Slow responding targets may disconnect to permit more efficient (faster) devices to be accessed while they prepare for the next data phase, or a target may disconnect if it recognizes that the next data phase in a burst transfer is out of its address range. A target disconnects by asserting STOP#, TRDY#, and DEVSEL# as shown in Figures 10a and 10b. The initiator in Figure 10a responds to the disconnect condition by deasserting FRAME# on the following clock but does not complete the data transfer until IRDY# is asserted. This situation can only occur when the S5933 is a target. When the S5933 is an initiator, IRDY# is always asserted during the data phase (no initiator wait states). The timing diagram in Figure 10b applies to the S5933 as either a target disconnecting or an initiator with its target performing a disconnect. The S5933 performs a target disconnect if a burst access is attempted to the PCI Operation Registers.

Figure 10a. Target Disconnect Example 1 (IRDY# deasserted)

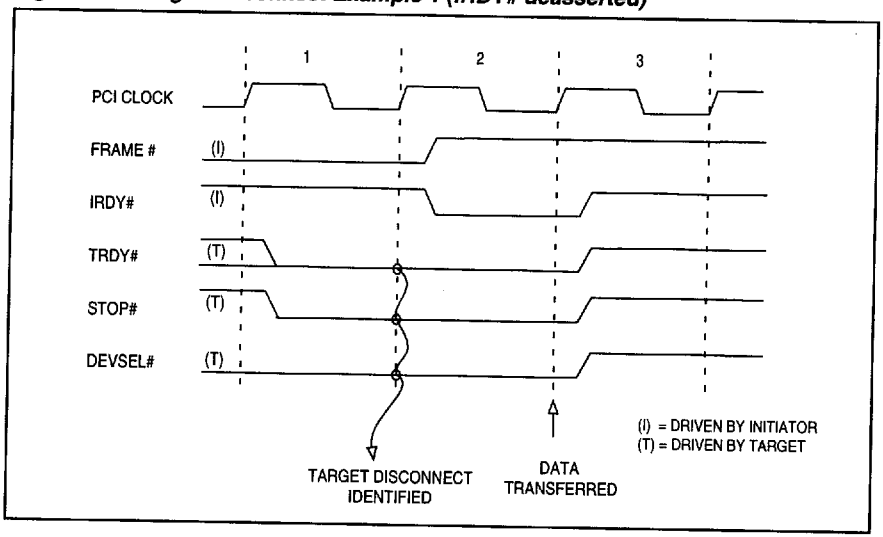
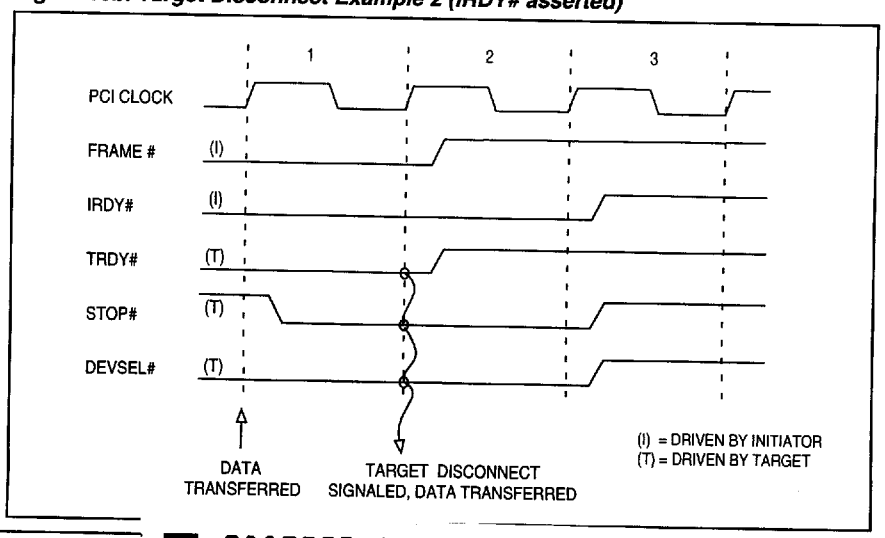


Figure 10b. Target Disconnect Example 2 (IRDY# asserted)



Target Requested Retries

When the S5933 FIFO registers are accessed (S5933 as a target) and data is unavailable (empty FIFO) for read transfers or cannot be accepted for write transfers (full FIFO), the S5933 immediately terminates the cycle by requesting a retry. The S5933 also initiates a retry for Pass-Thru writes where the Add-On has not completed the preceding Pass-Thru write by asserting PTRDY#, and for Pass-Thru reads where the Add-On cannot supply data within 8 PCI clocks (16 clocks for the first data phase of a burst). A retry is requested by a target asserting both STOP# and DEVSEL# while TRDY# is deasserted. Figure 11 shows the behavior of the S5933 when performing a target-initiated retry.

Target Aborts

A target abort termination represents an error condition where no number of retries will produce a successful target access. A target abort is uniquely identified by the target deasserting DEVSEL# and TRDY# while STOP# is asserted. When a target performs an abort, it must also set bit 11 of its PCI Status register. The S5933 configuration and operation registers never respond with a target abort when accessed. If the S5933 encounters this condition when operating as a PCI initiator, the S5933 sets bit 12 (received target abort) in the PCI Status register. Figure 12 depicts a target abort cycle.

Target termination types are summarized in Table 2.

Figure 11. Target-Initiated Retry

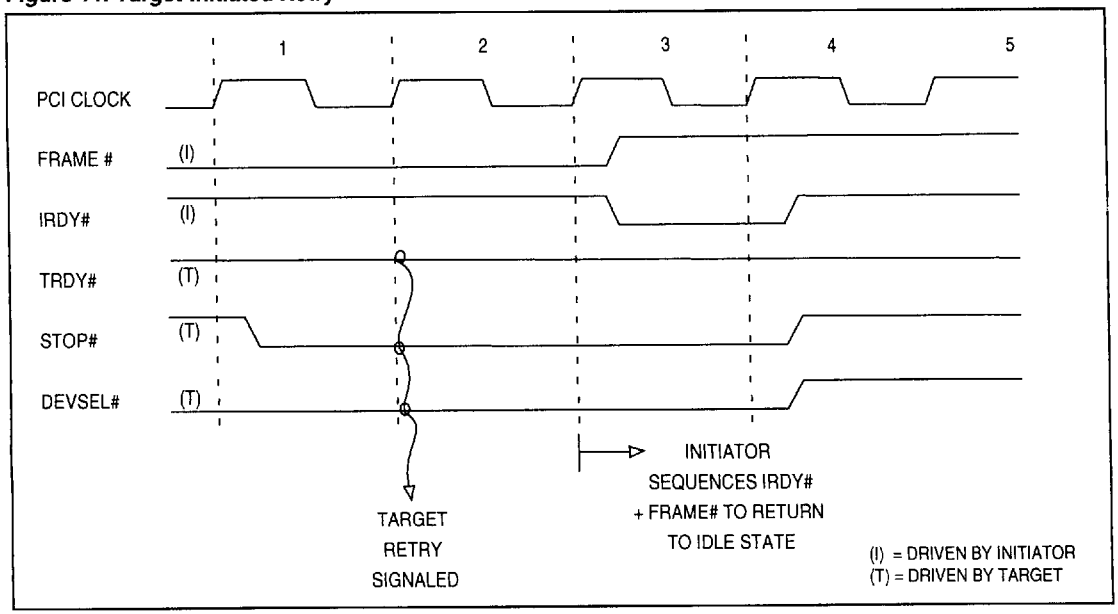


Table 2. Target Termination Types

Termination	DEVSEL#	STOP#	TRDY#	Comment
Disconnect	on	on	on	Data is transferred. Transaction needs to be re-initiated to complete.
Retry	on	on	off	Data was not transferred. Transaction should be tried later.
Abort	off	on	off	Data was not transferred. Fatal error.

Figure 12. Target Abort Example

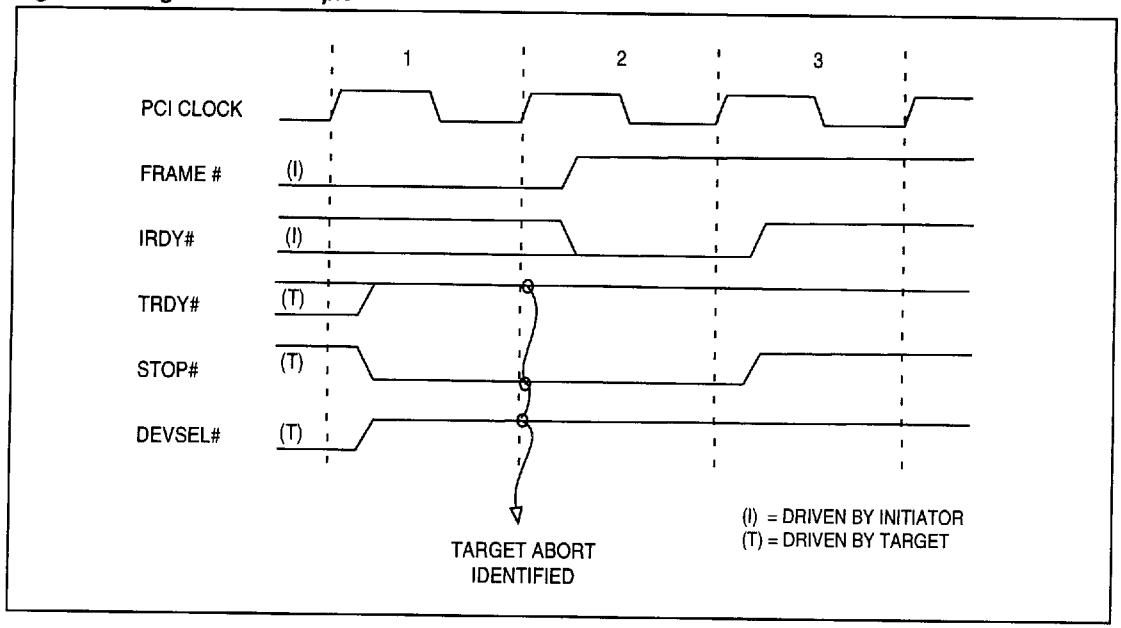
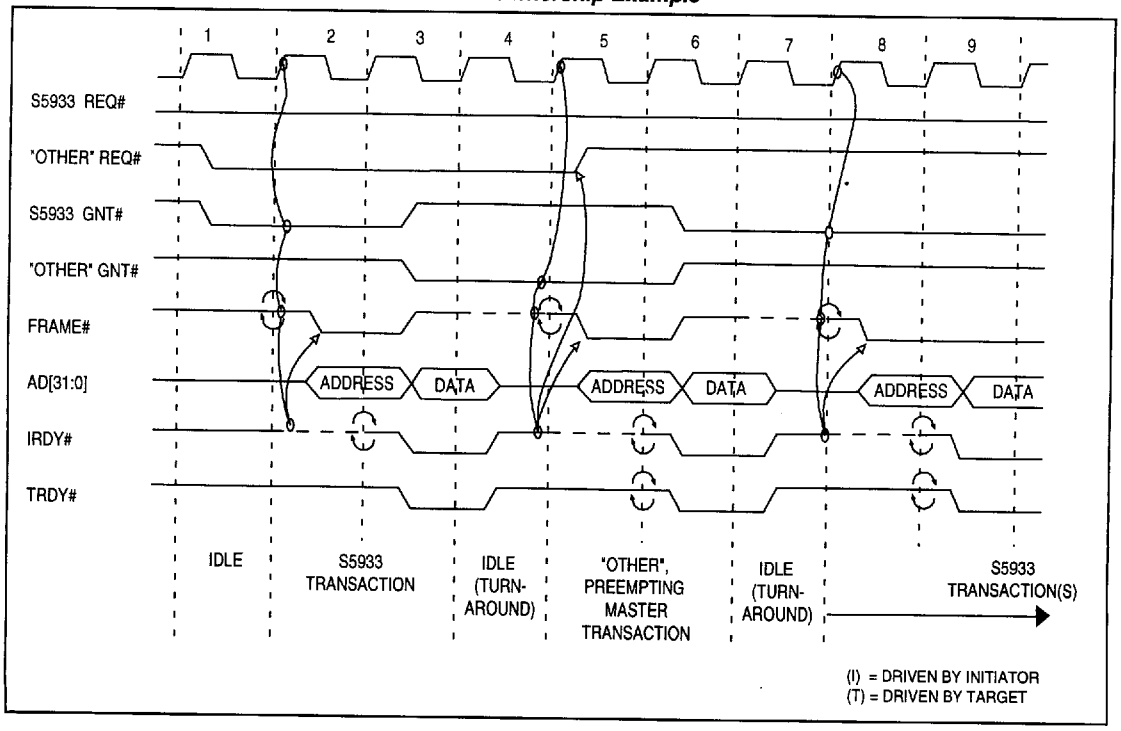


Figure 13. PCI Bus Arbitration and S5933 Bus Ownership Example



PCI BUS MASTERSHIP

When the S5933 requires PCI bus mastership, it presents a request via the REQ# signal. This signal is connected to the system's PCI bus arbiter.

Only one initiator (bus master) may control the PCI bus at a given time. The bus arbiter determines which initiator is given control of the bus. Control is granted to a requesting device by the arbiter asserting that device's grant signal (GNT#). Each REQ#/GNT# signal pair is unique to a given PCI agent.

After asserting REQ#, the S5933 assumes bus ownership on the first PCI clock edge where its GNT# input is asserted along with FRAME# and IRDY# deasserted (indicating no other device is generating PCI bus cycles). Once ownership is established by the S5933, it maintains ownership as long as the arbiter keeps its GNT# asserted. If GNT# is deasserted, the S5933 completes the current transaction. The S5933 does this by deasserting FRAME# and then deasserting IRDY# upon data transfer. Figure 13 shows a sequence where the S5933 is granted ownership of the bus and then is preempted by another master before the S5933 can finish its current transaction.

Bus Mastership Latency Components

It is often necessary for system designers to predict and guarantee that a minimum data transfer rate is sustainable to support a particular application. In the design of a bus mastering application, knowledge of the maximum delay a device might encounter from the time it requests the PCI bus to the time in which it is actually granted the bus is desirable. This allows the design to provide adequate data buffering. The PCI specification refers to this bus request to grant delay as "arbitration latency."

Once a PCI initiator has been granted the bus, the PCI specification defines the delay from the grant to the new initiator's assertion of FRAME# as the "bus acquisition latency." Afterwards, the delay from FRAME# asserted to target ready (TRDY#) asserted is defined as "target latency." Figure 14 shows a time-line depicting the components of PCI bus access latency.

There are numerous configuration variations possible with the PCI specification. A system designer can determine whether a bus master can support a critical, timely transfer by establishing a specific configuration and by defining these latency values. The S5933, as an initiator, produces the fastest response allowable for its bus acquisition latency (GNT# to FRAME# asserted). The S5933 also implements the PCI Master Latency Timer. Once granted the bus, the S5933 is guaranteed ownership for a minimum amount of time defined by the Master Latency Timer. The S5933, as an initiator, cannot control the responsiveness of a particular target nor the bus arbitration delay.

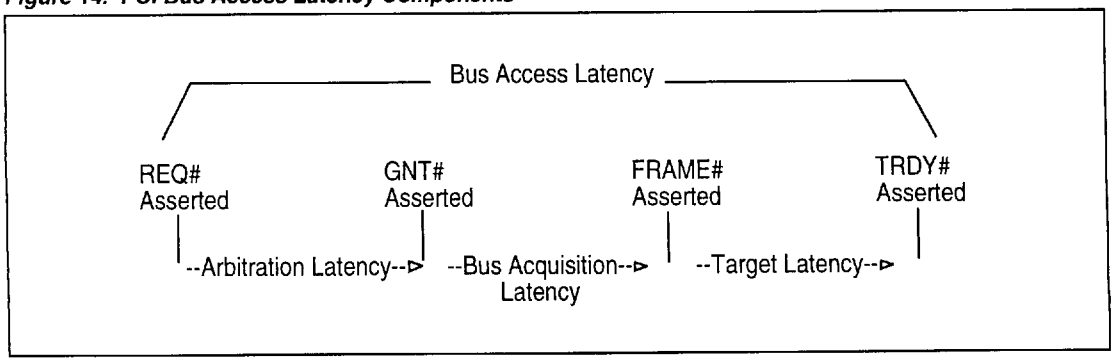
The PCI specification provides two mechanisms to control the amount of time a master may own the bus. One mechanism is through the master (master-initiated termination). The other is by the target and is achieved through a target-initiated disconnect.

Bus Arbitration

Although the PCI specification defines the condition that constitutes bus ownership, it does not provide rules to be used by the system's PCI bus arbiter in deciding which master is to be granted the PCI bus next. The arbitration priority scheme implemented by a system may be fixed, rotational, or custom. The arbitration latency is a function of the system, not the S5933.

3

Figure 14. PCI Bus Access Latency Components



Bus Acquisition

Once GNT# is asserted, giving bus ownership to the S5933, the S5933 must wait until the PCI bus becomes idle. This delay is called bus acquisition latency and involves the state of the signals FRAME# and IRDY#. The current bus master must complete its current transaction before the S5933 may drive the bus. Table 3 depicts the four possible combinations of FRAME# and IRDY# with their interpretation.

Target Latency

The PCI specification requires that a selected target relinquish the bus should an access to that target require more than eight PCI clock periods (16 clocks for the first data phase in a burst). Slow targets can exist within the PCI specification by using the target initiated retry. This prevents slow target devices from potentially monopolizing the PCI bus and also allows more accurate estimations for bus access latency.

Target Locking

It is possible for a PCI bus master to obtain exclusive access to a target ("locking") through use of the PCI bus signal LOCK#. LOCK# is different from the other PCI bus signals because its ownership may belong to any bus master, even if it does not currently have ownership of the PCI bus. The ownership of LOCK#, if not already claimed by another master, may be achieved by the current PCI bus master on the clock period following the initial assertion of FRAME#. Figure 15 describes the signal relationship for establishing a lock. The ownership of LOCK#, once established, persists even while other bus masters control the bus. Ownership can only be relinquished by the master which originally established the lock.

Figure 15. Engaging the LOCK# Signal

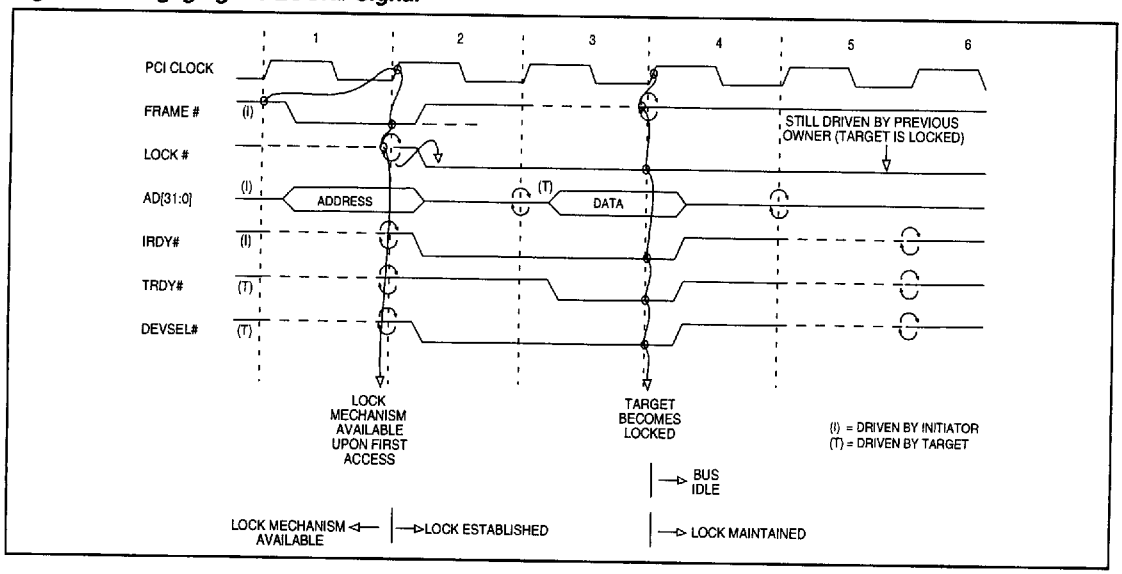


Table 3. Possible Combinations of FRAME# and IRDY#

FRAME#	IRDY#	Description
deasserted	deasserted	Bus Idle
deasserted	asserted	The initiator is ready to complete the last data transfer of a transaction.
asserted	deasserted	An Initiator has a transaction in progress but is not able to complete the data transfer on this clock.
asserted	asserted	An initiator has a transaction in progress and is able to complete a data transfer.

Targets selected with LOCK# deasserted during the assertion of FRAME# (clock period 1 of Figure 15), which encounter the assertion of LOCK# during the following clock (clock period 2 of Figure 15) are thereafter considered "locked." A target, once locked, requires that subsequent accesses to it deassert LOCK# while FRAME# is asserted. Figure 16 show a subsequent access to a locked target by the master which locked it. Because LOCK# is owned by a single master, only that master is able to deassert it at the beginning of a transaction (allowing successful access to the locked target). A locked target can only be unlocked during the clock period following the last data transfer of a transaction when the LOCK# signal is deasserted.

An unlocked target ignores LOCK# when it observes that LOCK# is already asserted during the first clock period of a transaction. This allows other masters to access other (unlocked) targets. If an access to a locked target is attempted by a master other than the one that locked it, the target responds with a retry request, as shown in Figure 17.

The S5933 responds to and supports bus masters which lock it as a target. When the S5933 is a bus master, it never attempts to lock a target, but it honors a target's request for retry if that target is locked by another master.

Figure 16. Access to a Locked Target by its Owner

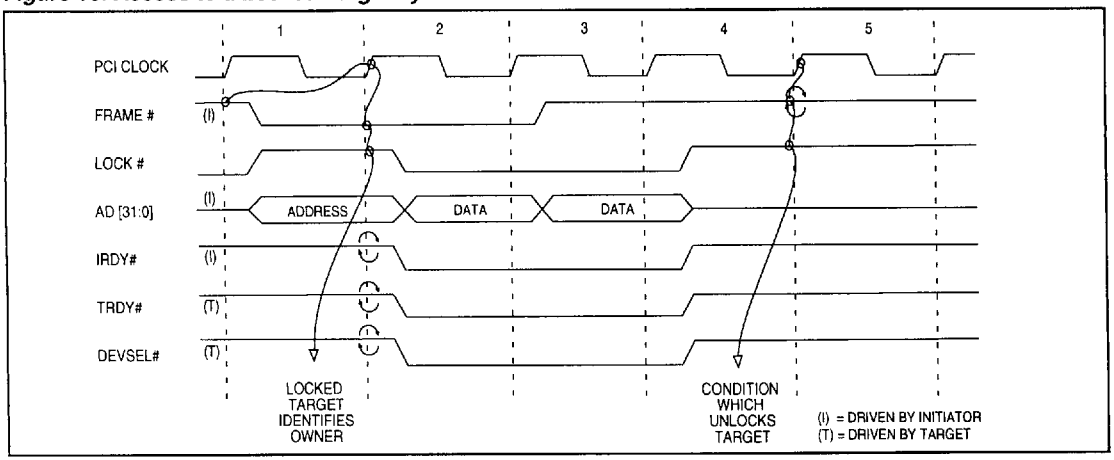
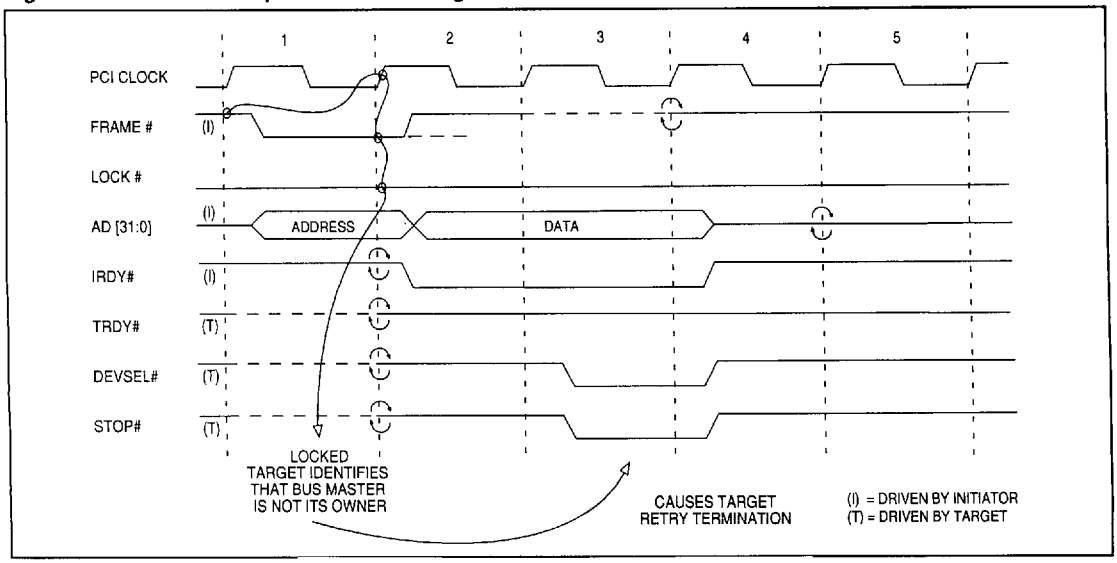


Figure 17. Access Attempt to a Locked Target



PCI BUS INTERRUPTS

The S5933 controller is able to generate PCI bus interrupts by asserting the PCI bus interrupt signal (INTA#). INTA# is a multisourced, wire-ORed signal on the PCI bus and is driven by an open drain output on the S5933. The assertion and deassertion of INTA# have no fixed timing relationship with respect to the PCI bus clock. Once the S5933 asserts INTA#, it remains asserted until the interrupt source is cleared by a write to the Interrupt Control/Status Register (INTCSR).

PCI BUS PARITY ERRORS

The PCI specification defines two error-reporting signals, PERR# and SERR#. These signals indicate a parity error condition on the signals AD[31:0], C/BE[3:0]#, and PAR. The validity of the PAR signal is delayed one clock period from its corresponding AD[31:0] and C/BE[3:0]# signals. Even parity exists when the total number of ones in the group of signals is equal to an even number. PERR# is the error-reporting mechanism for parity errors that occur during the data phase for all but PCI Special Cycle commands. SERR# is the error-reporting mechanism for parity errors that occur during the address phase.

The timing diagram in Figure 18 shows the timing relationships between the signals AD[31:0], C/BE[3:0]#, PAR, PERR# and SERR#.

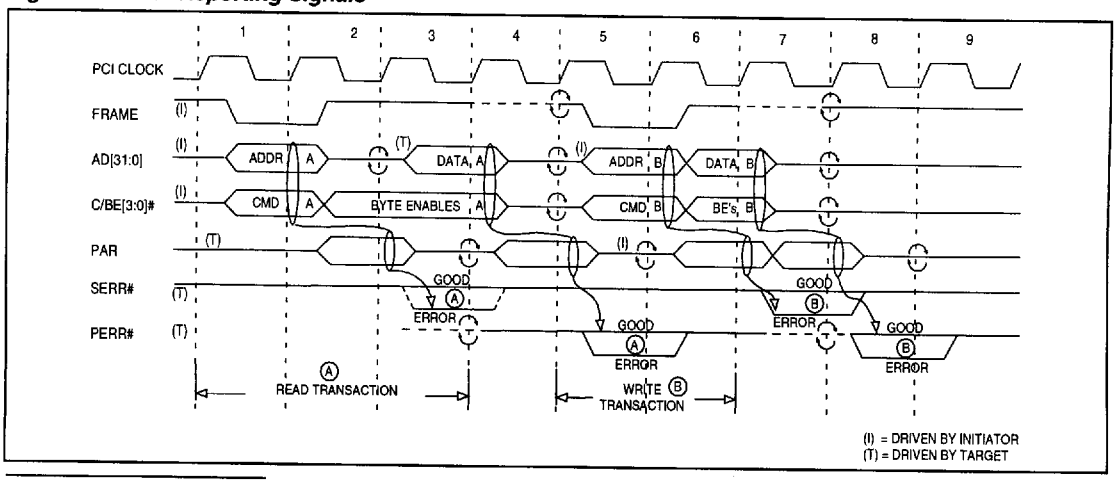
The S5933 asserts SERR# if it detects odd parity during an address phase, if enabled. The SERR# enable bit is bit 8 in the S5933 PCI Command Register. The odd parity error condition involves the state of signals AD[31:0] and C/BE[3:0]# when FRAME# is first asserted and the PAR signal during the following clock. If an error is detected, the S5933 asserts SERR# on the following (after PAR valid) clock.

Since many targets may observe an error on an address phase, the SERR# signal is an open drain multisourced, wire-ORed signal on the PCI bus. The S5933 drives SERR# low for one clock period when an address phase error is detected. Once an SERR error is detected by the S5933, the PCI Status register bit 14, System Error, is set and remains until cleared through software or a hardware reset.

The PERR# signal is similar to the SERR# with two differences: it reports errors for the data phase and is only asserted by the device receiving the data. The S5933 drives this signal (removed from tri-state) when it is the selected target for write transactions or when it is the current master for bus read transactions. The parity error conditions are only reflected by the PERR# pin if the Parity Error Enable bit (bit 6) of the PCI Command register is set. Upon the detection of a data parity error, the Detected Parity Error bit (bit 15) of the PCI Status Register is set. Unlike the PERR# signal pin, this Status bit sets regardless of the state of the PCI Command register Parity Error Enable bit. An additional status bit (bit 8) called "Data Parity Reported" of the PCI Status register is employed to report parity errors that occur when the S5933 is the bus master. The "Data Parity Error Reported" status requires that the Parity Error Enable bit be set in the PCI Command register.

The assertion of PERR# occurs two clock periods following the data transfer. This two-clock delay occurs because the PAR signal does not become valid until the clock following the transfer, and an additional clock is provided to generate and assert PERR# once an error is detected. PERR# is only asserted for one clock cycle for each error sensed. The S5933 only qualifies the parity error detection during the actual data transfer portion of a data phase (when both IRDY# and TRDY# are asserted).

Figure 18. Error Reporting Signals



ADD-ON BUS INTERFACE

This chapter describes the Add-On bus interface for the S5933. The S5933 is designed to support connection to a variety of microprocessor buses and/or peripheral devices. The Add-On interface controls S5933 operation through the Add-On Operation Registers. These registers act as the Pass-Thru, FIFO, non-volatile memory and mailbox interfaces as well as offering control and status information.

Depending on the register being accessed, the interface may be synchronous, asynchronous, or configurable. To enhance performance and simplify Add-On logic design, some registers allow direct access with a single device input pin. The following sections describe the various interfaces to the PCI bus and how they are accessed from the Add-On interface.

ADD-ON OPERATION REGISTER ACCESSES

The S5933 Add-On bus interface is very similar to that of a memory or peripheral device found in a microprocessor-based system. A 32-bit data bus with individual read and write strobes, a chip enable and byte enables are provided. Other Add-On interface signals are provided to simplify Add-On logic design.

Accesses to the S5933 registers are done synchronous or asynchronous to BPCLK. For S5933 functions that are compatible with an Add-On microprocessor interface, it is helpful to allow an asynchronous interface, as the processor may not operate at the PCI bus clock frequency.

Add-On Interface Signals

The Add-On interface provides a small number of system signals to allow the Add-On to monitor PCI bus activity, indicate status conditions (interrupts), and allow Add-On bus configuration. A standard bus interface is provided for Add-On Operation Register accesses.

System Signals

BPCLK and SYSRST# allow the Add-On interface to monitor the PCI bus status. BPCLK is a buffered version of the PCI clock. The PCI clock can operate from 0 MHz to 33 MHz. SYSRST# is a buffered version of the PCI reset signal, and may also be toggled by host application software through bit 24 of the Bus Master Control/Status Register (MCSR).

IRQ# is the Add-On interrupt output. This signal is active low and can indicate a number of conditions. Add-On interrupts may be generated from the mailbox or FIFO interfaces. The exact conditions which generate an interrupt are discussed in the mailbox and FIFO chapters. The interrupt output is deasserted when acknowledged by an access to the Add-On Interrupt Control/Status Register (AINT). All interrupt sources are cleared by writing a one to the corresponding interrupt bit.

The MODE input on the Add-On interface configures the datapath width for the Add-On interface. MODE low indicates a 32-bit data bus. MODE high indicates a 16-bit data bus. For 16-bit operation, BE3# is redefined as ADR1, providing an extra address input. ADR1 selects the low or high words of the 32-bit S5933 Add-On Operation Registers.

Register Access Signals

Simple register accesses to the S5933 Add-On Operation Registers take two forms: synchronous to BPCLK and asynchronous. The following signals are required to complete a register access to the S5933.

BE[3:0]# Byte Enable Inputs. These S5933 inputs identify valid byte lanes during Add-On transactions. When MODE is set for 16-bit operation, BE2# is not defined and BE3# becomes ADR1.

ADR[6:2] Address Inputs. These address pins identify the specific Add-On Operation Register being accessed. When configured for 16-bit operation (MODE=1), an additional input, ADR1 is available to allow the 32-bit operation registers to be accessed with two 16-bit cycles.

RD# Read Strobe Input.

WR# Write Strobe Input.

SELECT# Chip Select Input. This input identifies a valid S5933 access.

DQ[31:0] Bidirectional Data Bus. These I/O pins are the S5933 data bus. When configured for 16-bit operation, only DQ[15:0] are valid.

In addition, there are dedicated signals for FIFO accesses (RDFIFO# and WRFIFO#) and Pass-Thru address accesses (PTADR#). These are discussed separately in the FIFO and Pass-Thru sections of this chapter.

The internal interfaces of the S5933 allow Add-On Operation Registers to be accessed asynchronous to BPCLK (synchronous to the rising edge of the read or write strobe). The exception to this is the Add-On General Control/Status Register. This is due to the async nature of FIFO status bits changing as the PCI bus reads data. For Pass-Thru operations, the Pass-Thru Data Register accesses are synchronous to BPCLK to support burst transfers. The FIFO port may also be accessed synchronous to BPCLK, if configured to do so.

Asynchronous Register Accesses

For many Add-On applications, Add-On logic does not operate at the PCI bus frequency. This is especially true for Add-Ons implementing a microprocessor, which may be operating at a lower (or higher) frequency. Figures 1 and 2 show asynchronous Add-On Operation Register accesses. Exact AC timings are detailed in the Electrical and AC Characteristics chapter (Chapter 13).

For asynchronous reads (Figure 1), data is driven on the data bus when RD# is asserted. When RD# is not asserted, the DQ[31:0] outputs float. A valid address and valid byte enables must be presented before correct data is driven. RD# has both a minimum inactive time and a minimum active time for asynchronous accesses.

For asynchronous writes (Figure 2), data is clocked into the S5933 on the rising edge of the WR# input. Address, byte enables, and data must all meet setup and hold times relative to the rising edge or WR#. WR# has both a minimum inactive time and a minimum active time for asynchronous accesses

Synchronous FIFO and Pass-Thru Data Register Accesses

To obtain the highest data transfer rates possible, Add-On logic should operate synchronously with the PCI clock. The buffered PCI clock (BPCLK) is provided for this purpose. A synchronous interface with Pass-Thru mode or the FIFO allows data to be transferred at the maximum PCI bus bandwidth (132 MBytes/sec) by allowing burst accesses with the Add-On interface. The RD# and WR# inputs become enables, using BPCLK to clock data into and out of registers. This section applies only to synchronous accesses to the FIFO (AFIFO) and Pass-Thru Data (APTD) registers.

Figures 3 and 4 show single-cycle, synchronous FIFO and Pass-Thru Operation Register accesses. Exact AC timings are detailed in the Electrical and AC Characteristics chapter.

For synchronous reads (Figure 3), data is driven onto the data bus when RD# (or RDFIFO#) is asserted. When RD# is not asserted, the DQ[31:0] outputs float. The address, byte enable, and RD# inputs must meet setup and hold times relative to the rising edge of BPCLK. Burst reads may be performed by holding RD# low.

For synchronous writes (Figure 4), data is clocked into the register on the rising edge of BPCLK. Address, byte enables, and data must all meet setup and hold times relative to the rising edge or BPCLK. Burst writes may be performed by holding WR# (or WRFIFO#) low. When holding WR# low, data is clocked in on each BPCLK rising edge.

nv Memory Accesses Through the Add-On General Control/Status Register

To access nv memory contents through the Add-On General Control/Status Register (AGCSTS), special considerations must be made. Internally, all nv memory accesses by the S5933 are synchronized to a divided-down version of the PCI bus clock. Because of this, if nv memory accesses are performed through the AGCSTS register, the register access must be synchronized to BPCLK. The rising edge RD# or WR# is still used to clock data, but these inputs along with the address and byte enables are synchronized to BPCLK. Accesses to AGCSTS for monitoring FIFO or mailbox status, etc., may be done asynchronous to BPCLK.

MAILBOX BUS INTERFACE

The mailbox register names may need some clarification. For the Add-On interface, an outgoing mailbox refers to a mailbox sending information to the PCI bus. An incoming mailbox refers to a mailbox receiving information from the PCI bus. An outgoing mailbox on the Add-On interface is, internally, the same as the corresponding incoming mailbox on the PCI interface and vice-versa.

Figure 1. Asynchronous Add-On Operation Register Read

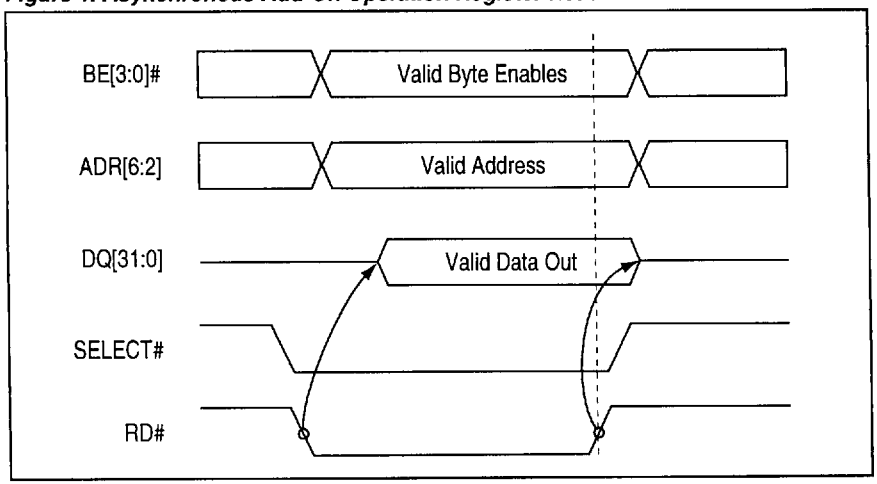
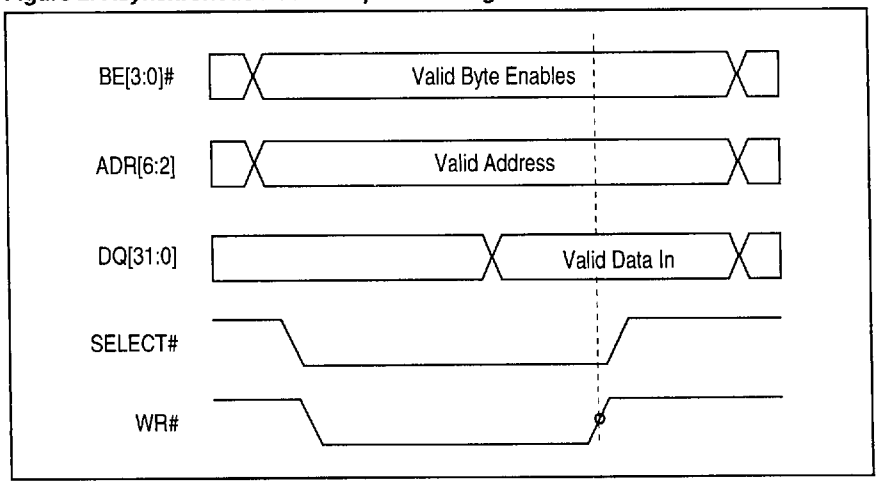


Figure 2. Asynchronous Add-On Operation Register Write



3

Figure 3. Synchronous FIFO or Pass-Thru Data Register Read

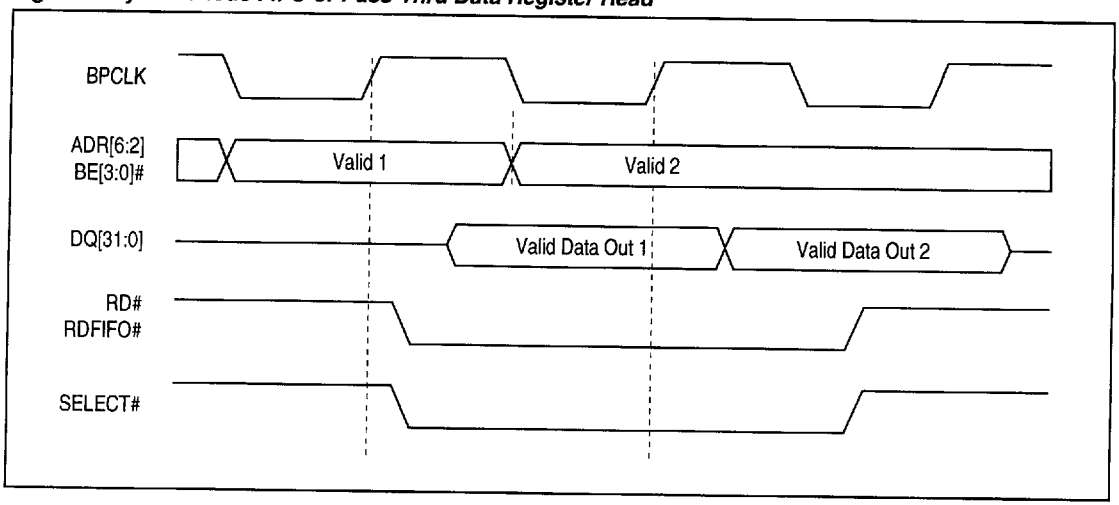
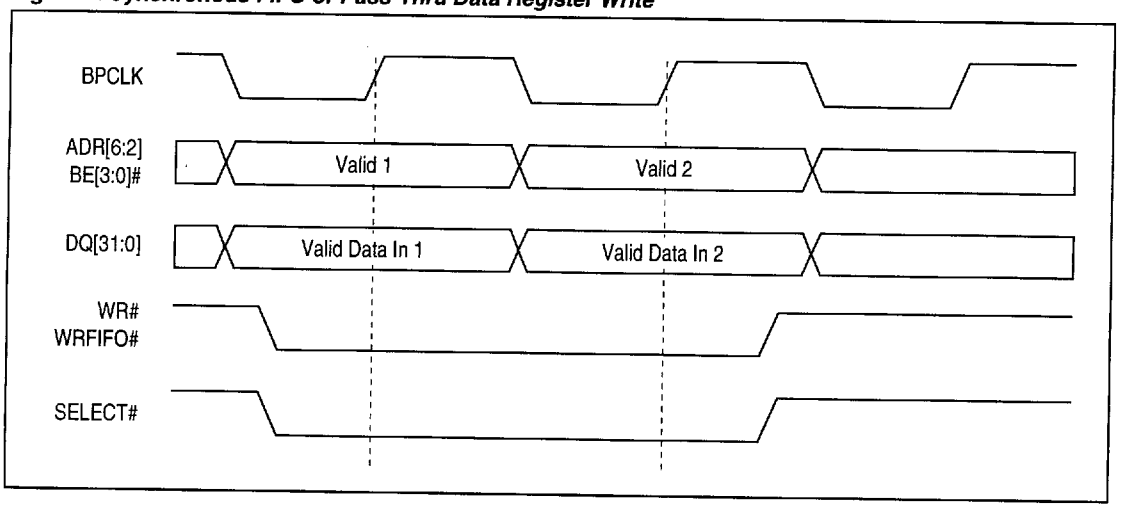


Figure 4. Synchronous FIFO or Pass-Thru Data Register Write



Mailbox Interrupts

Mailboxes can be configured to generate Add-On interrupts (IRQ#) and/or allow the Add-On to generate PCI interrupts (INTA#). Mailbox empty/full status conditions be can used to interrupt the Add-On or PCI host to indicate some action is required. An individual mailbox byte is selected to generate an interrupt when accessed. An outgoing mailbox becoming empty or an incoming mailbox becoming full asserts the interrupt output (if enabled).

When used with a serial nv memory boot device, the mailboxes also provide a way to generate PCI interrupts (INTA#) through hardware. When a serial nv memory boot device is used, the device pin functions EA0 - EA8 are redefined. These pins then provide direct, external access to the Add-On outgoing mailbox 4, byte 3 (which is also PCI incoming mailbox 4, byte 3).

FIFO BUS INTERFACE

The FIFO register on the Add-On interface may be accessed synchronously or asynchronously.Location 45h, bits 6 and 5 in the nv memory boot device determine the interface method. If no boot device is used, the default condition is an asynchronous FIFO interface.

FIFO Direct Access Inputs

RDFIFO# and WRFIFO# are referred to as FIFO 'direct access' inputs. Asserting RDFIFO# is functionally identical to accessing the FIFO with RD#, SELECT#, BE[3:0]#, and ADR[6:2]. Asserting WRFIFO# is functionally identical to accessing the FIFO with WR#, SELECT#, BE[3:0]#, and ADR[6:2]. RD# and WR# must be deasserted when RDFIFO# or WRFIFO# is asserted, but SELECT# may be asserted. These inputs automatically drive the address (internally) to 20h and assert all byte enables. The ADR[6:2] and BE[3:0]# inputs are ignored when using the FIFO direct access inputs. RDFIFO# and WRFIFO# are useful for Add-On designs which cascade an external FIFO into the S5933 FIFO or use dedicated external logic to access the FIFO.

Direct access signals always access the FIFO as 16-bits or 32-bits, whatever the MODE pin is configured for. For 16-bit mode, two consecutive accesses fill or empty the 32-bit FIFO register.

FIFO Status Signals

The FIFO Status signals indicate to the Add-On logic the current state of the S5933 FIFO. A FIFO status change caused by a PCI FIFO access is reflected one PCI clock period after the PCI access is completed (TRDY# asserted). A FIFO status change caused by an Add-On FIFO access is reflected immediately (after a short propagation delay) after the access occurs. For Add-On accesses, FIFO status is updated after the rising edge of BPCLK for synchronous interfaces or after the rising edge of the read or write strobe for asynchronous interfaces.

FIFO Control Signals

For Add-On initiated PCI bus mastering, the FIFO status reset controls FWC# (Add-On to PCI FIFO clear) and FRC# (PCI to Add-On FIFO clear) are available. FWC# and FRC# must be asserted for a minimum of one BPCLK period to be recognized. These inputs are sampled at the rising edge of BPCLK. These inputs should not be asserted unless the FIFO is idle. Asserting a FIFO status reset input during a PCI or Add-On FIFO access results in indeterminate operation.

For Add-On initiated bus master transfers, AMREN (Add-On bus master read enable) and AMWEN (Add-On bus master write enable) are used, in conjunction with the appropriate FIFO status signals, to enable the S5933 to assert its PCI bus request (REQ#).

PASS-THRU BUS INTERFACE

The S5933 Pass-Thru interface is synchronous. The Add-On Pass-Thru Address (APTA) and Add-On Pass-Thru Data (APTD) registers may be accessed pseudo-synchronously.

Although BPCLK is used to clock data into and out of the Pass-Thru registers, accesses may be performed asynchronously. For reads, APTA or APTD data remains valid as long as RD# (or PTADR#) is asserted. A new value is not driven until PTRDY# is asserted by Add-On logic. For writes to APTD, data is clocked into the S5933 on every BPCLK rising edge, but is not passed to the PCI bus until PTRDY# is asserted. PTRDY# must by synchronized to BPCLK.

Pass-Thru Status Indicators

The Pass-Thru status indicators indicate that a Pass-Thru access is in process and what action is required by the Add-On logic to complete the access. All Pass-Thru status indicators are synchronous with the PCI clock.

Pass-Thru Control Inputs

Some Pass-Thru implementations may require an address corresponding to the Pass-Thru data. The Add-On Pass-Thru Address Register (APTA) contains the PCI address for the Pass-Thru cycle. To allow access to the Pass-Thru address without generating an Add-On read cycle, PTADR# is provided. PTADR# is a direct access input for the Pass-Thru address. Asserting PTADR# is functionally identical to accessing the Pass-Thru address register with RD#, SELECT#, BE[3:0]#, and ADR[6:2]. RD# and WR# must be deasserted when PTADR# is asserted, but SELECT# may be asserted. These inputs automatically drive the address (internally) to 28h and assert all byte enables. The ADR[6:2] and BE[3:0]# are ignored when using the PTADR# direct access input. When PTADR# is asserted, the contents of the APTA register are immediately driven onto the Add-On data bus.

The PTADR# direct access signal accesses the Pass-Thru address register as 16-bits or 32-bits, whatever the MODE pin is configured for. For 16-bit mode, PTADR# only presents the lower 16-bits of the APTA register.

PTRDY# indicates that the Add-On has completed the current Pass-Thru access. Multiple Add-On reads or writes may occur to the Pass-Thru data (APTD) register before asserting PTRDY#. This may be required for 8-bit or 16-bit Add-On interfaces using multiple accesses to the 32-bit Pass-Thru data register. In some cases, the Add-On bus may be 32-bits, but logic may require multiple BPCLK periods to read or write data. In this situation, accesses may be extended by holding off PTRDY#. PTRDY# must be synchronized to BPCLK.

NON-VOLATILE MEMORY INTERFACE

The S5933 allows read and write access to the nv memory device used for configuration. Reads are necessary during device initialization as configuration information is downloaded into the S5933. If an expansion BIOS is implemented in the nv memory, the host transfers (shadows) the code into system DRAM. Writes are useful for in-field updates to expansion BIOS code. This allows software to update the nv memory contents without altering hardware.

Non-Volatile Memory Interface Signals

For serial nv memory devices, there are only two signals used to interface with nv memory. SCL is the serial clock, and SDA is the serial data line. The functionality of these signals is described in-detail in the PIN description Section of this book. The designer does not need to generate the timings for SCL and SDA. The S5933 automatically performs the correct serial access when programmed for for serial devices.

For byte-wide nv memory devices, there is an 8-bit data bus (EQ7:0), and a 16-bit address bus (EA15:0) dedicated for the nv memory interface. When a serial nv memory is implemented, many of these pins have alternate functions. The S5933 also has read (ERD#) and write (EWR#) outputs to drive the OE# and WR# inputs on a byte-wide nv memory. The designer does not need to generate the timings for these outputs. The S5933 automatically performs the read and write accesses when programmed for byte wide devices.

Accessing Non-Volatile Memory

The nv memory, if implemented, can be accessed through the PCI interface or the Add-On interface. Accesses from both the PCI side and the Add-On side must be synchronous with the PCI clock (BPCLK for the Add-On). Accesses to the nv memory from the PCI interface are through the Bus Master Control/Status Register (MCSR) PCI Operation Register. Accesses to the nv memory from the Add-On interface are through the Add-On General Control/Status Register (AGCSTS) Add-On Operation Register.

Accesses to the MCSR register are from the PCI bus and are, therefore, automatically synchronous to the PCI clock. Accesses to the AGCSTS register from the Add-On side must be synchronous with respect to BPCLK.

Some nv memories may contain Expansion ROM BIOS code for use by the host software. During initialization, the Expansion BIOS is located within system memory. The starting location of the nv memory is stored in the Expansion ROM Base Address Register in the S5933 PCI Configuration Registers. A PCI read from this region results in the S5933 performing four consecutive byte access to the nv memory device. Writes to the nv memory are not allowed by writing to this region. Writes to the nv memory must be performed as described below.

The S5933 contains two latches within the MCSR register to control and access the NVRAM. One is an 8 bit latch called the NVRAM Address/Data Register which is used to hold NVRAM address and data information. The other is a 3 bit latch called the NVRAM Access Control Register which is used to direct the address and data information and to control the NVRAM itself. Reading or writing to the NVRAM is performed through bits D31:29 of this register. These bits are enable and decode controls rather than a command or instruction to be executed. D31 of this register is the primary enable bit which allows all accesses to occur. When written to a '1', D31 enables the decode bits D30 and D29 to direct the data contained in the address/data latch, D23:16, to the low address, high address or data latches. D31 should be thought of as "opening a door" where as long as D31 = 1, then the door is open for address or data information to be altered. The table on page 5-16 of the S5933 data book shows the D31:29 bit combinations for reading, writing, and loading address/data information. Additionally, D31 doubles as an S5933 status bit. A '1' indicates that the S5933 is currently busy reading or writing to the NVRAM. A '0' indicates a complete or inactive state.

For the examples below, we will assume the S5933 is I/O mapped with a base address of FC00h. These examples will read one byte of the Vendor ID and write one byte to the Vendor ID.

This example will write 1 byte from NVRAM location 0040h and read it back:

In	**FC00h + 3Fh** (offset of NVRAM Access Control Register) until **D31 = 0** (not busy).
Out	**FC00h + 3FH** an **80h** (CMD to load the low address byte). This sets decode bits and opens door for low address latch.
Out	**FC00h + 3Eh** (offset of Address/Data Register) **40h** (the low byte of the address desired) 40h goes into latch but is not latched yet.
Out	**FC00h + 3Fh** an **A0h** (CMD to load the high address byte). This latches the low address through changing the decode bits and opens the door for the high address latch.
Out	**FC00h + 3Eh** a **00h** (the high byte of the address desired). 00h goes into the latch but is not latched yet.
Out	**FC00h + 3Fh** an **00h** (inactive CMD). This latches the high address through the disabling D31, 'closes the door'.
Out	**FC00h + 3Eh DATA** (the data byte to be written). DATA byte goes into the latch but is not latched yet.
Out	**FC00h + 3Fh** a **C0h** (CMD to write the data byte). This latches the data byte through changing the decode bits and begins to write NVRAM data operation.
In	**FC00h + 3Fh** until **D31 = 0** (not busy).
Out	**FC00h + 3Fh** an **E0h** (CMD to read the address latched).
In	**FC00h + 3Fh** until **D31 = 0** (not busy).
In	**FC00h + 3Eh** the **data.**

This example will read 1 byte from NVRAM location 0040h:

In	**FC00h + 3Fh** (offset of NVRAM Access Control Register) until **D31 = 0** (not busy).
Out	**FC00h + 3Fh** an **80h** (CMD to load the low address byte). This sets decode bits and opens door for low address latch.
Out	**FC00h + 3Eh** (offset of Address/Data Register) **40h** (the low byte of the address desired) 40h goes into latch but is not latched yet.
Out	**FC00h + 3Fh** an **A0h** (CMD to load the high address byte). This latches the low address through changing the decode bits and opens the door for the high address latch.
Out	**FC00h + 3Eh** a **00h** (the high byte of the address desired) 00h goes into latch but is not latched yet.
Out	**FC00h + 3Fh** an **E0h** (CMD to read NVRAM data). This latches the high address through changing the decode bits and begins to read the NVRAM data operation.
In	**FC00h + 3Fh** until **D31 = 0** (not busy).
In	**FC00h + 3Eh** the **data.**

This example will read 1 byte from NVRAM location 0041h and contains an extra step to demonstrate D31 operation:

In	**FC00h + 3Fh** (offset of NVRAM Access Control Register) until **D31 = 0** (not busy).
Out	**FC00h + 3Fh** an **80h** (CMD to load the low address byte). This sets decode bits and opens the door for low address latch.
Out	**FC00h + 3Eh** (offset of Address/Data Register) **40h** (the low byte of the address desired) 40h goes into latch but is not latched yet.
Out	**FC00h + 3Eh** (offset of Address/Data Register) **41h** (the low byte of the address desired) 41h goes into latch but is not latched yet.
Out	**FC00h + 3Fh** an **A0h** (CMD to load the high address byte). This latches the low address through changing the decode bits and opens the door for the high address latch.
Out	**FC00h + 3Eh 00h** (the high byte of the address desired) 00h goes into latch but is not latched yet.
Out	**FC00h + 3Fh** an **E0h** (CMD to read the address latched). This latches the high address through changing the decode bits and begins the read NVRAM data operation.
In	**FC00h + 3Fh** until **D31 = 0** (not busy).
In	**FC00h + 3Eh** the **data.**

Notes:
1. Latched addresses do not automatically increment after a read or write. They must be loaded with new values.
2. Latched addresses remain after reads and writes. It is allowable to only update one address byte for the next access.
3. A processor may perform a one word write to load an address byte and control command simultaneously.

nv Memory Device Timing Requirements

For serial nv memory devices, the serial clock output frequency is the PCI clock frequency divided by 512. This is approximately 65 KHz (with a 33 MHz PCI clock). Any serial memory device that operates at this frequency is compatible with the S5933.

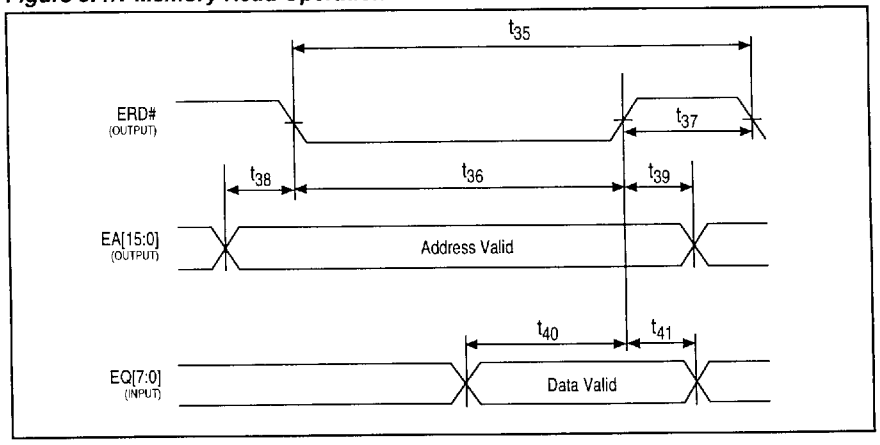
For byte-wide accesses, the S5933 generates the waveforms shown in Figures 5 and 6. Figure 5 shows an nv memory read operation. Figure 6 shows an nv memory write operation. Read operations are always the same length. Write operations, due to the characteristics of reprogrammable nv memory devices, may be controlled through a programming sequence.

Memory Device Requirements for Read Accesses

Timing	Spec.	T = 30 ns
Read cycle time	8T(max)	240 ns
Address valid to data valid	7T–10(max)	200 ns
Address valid to read active	T(max)	30 ns
Read active to data valid	6T–10(max)	170 ns
Read pulse width	6T(max)	180 ns
Data hold from read inactive	—	2 ns

3

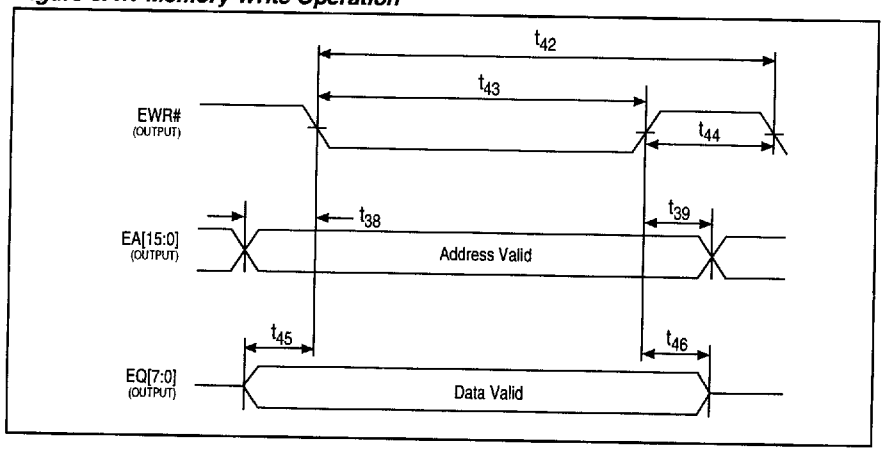
Figure 5. nv Memory Read Operation



Memory Device Requirements for Write Accesses

Timing	Spec.	T = 30 ns
Write cycle time	8T	Note 1
Address valid to write active	T(max)	30 ns
Data valid to write inactive	6T+10(max)	190 ns
Data hold from write inactive	T(max)	30 ns
Write pulse width	6T(max)	180 ns
Write inactive	Note 2	2 ns

Figure 6. nv Memory Write Operation



MAILBOX OVERVIEW

The S5933 has eight 32-bit mailbox registers. The mailboxes are useful for passing command and status information between the Add-On and the PCI bus. The PCI interface has four incoming mailboxes (Add-On to PCI) and four outgoing mailboxes (PCI to Add-On). The Add-On interface has four incoming mailboxes (PCI to Add-On) and four outgoing mailboxes (Add-On to PCI). The PCI incoming and Add-On outgoing mailboxes are the same, internally. The Add-On incoming and PCI outgoing mailboxes are also the same, internally.

The mailbox status may be monitored in two ways. The PCI and Add-On interfaces each have a mailbox status register to indicate the empty/full status of bytes within the mailboxes. The mailboxes may also be configured to generate interrupts to the PCI and/or Add-On interface. One outgoing and one incoming mailbox on each interface can be configured to generate interrupts.

FUNCTIONAL DESCRIPTION

Figure 1 shows a block diagram of the PCI to Add-On mailbox registers. Add-On incoming mailbox read accesses pass through an output interlock latch. This prevents a PCI bus write to a PCI outgoing mailbox from corrupting data being read by the Add-On. Figure 2 shows a block diagram of the Add-On to PCI mailbox registers. PCI incoming mailbox reads also pass through an interlocking mechanism. This prevents an Add-On write to an outgoing mailbox from corrupting data being read by the PCI bus. The following sections describe the mailbox flag functionality and the mailbox interrupt capabilities.

Figure 1. Block Diagram - PCI to Add-On Mailbox Register

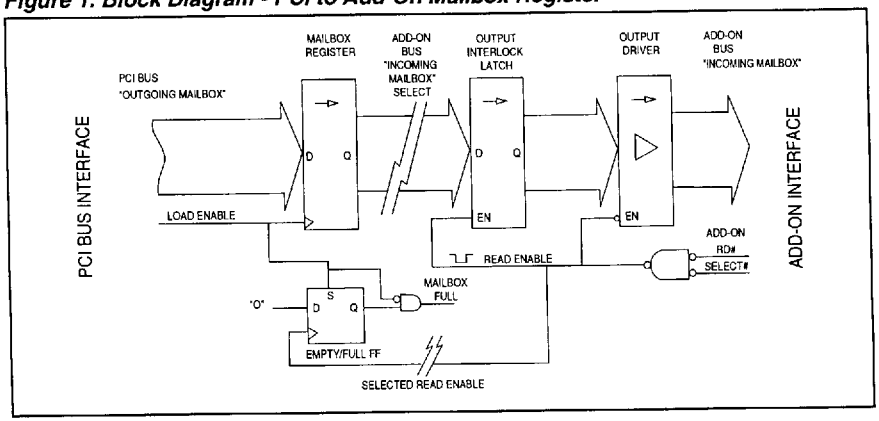
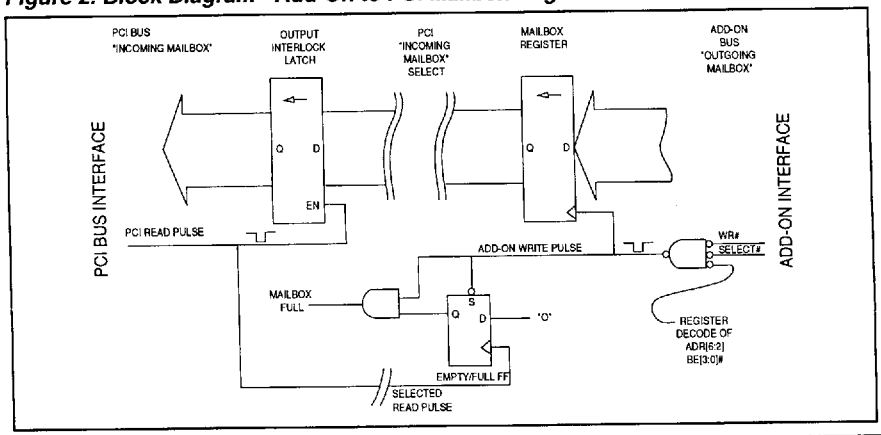


Figure 2. Block Diagram - Add-On to PCI Mailbox Register



Mailbox Empty/Full Conditions

The PCI and Add-On interfaces each have a mailbox status register. The PCI Mailbox Empty/Full Status (MBEF) and Add-On Mailbox Empty/Full Status (AMBEF) Registers indicate the status of all bytes within the mailbox registers. A write to an outgoing mailbox sets the status bits for that mailbox. The byte enables determine which bytes within the mailbox become full (and which status bits are set).

An outgoing mailbox for one interface is an incoming mailbox for the other. Therefore, incoming mailbox status bits on one interface are identical to the corresponding outgoing mailbox status bits on the other interface. The following list shows the relationship between the mailbox registers on the PCI and Add-On interfaces.

PCI Interface		Add-On Interface
Outgoing Mailbox 1	=	Incoming Mailbox 1
Outgoing Mailbox 2	=	Incoming Mailbox 2
Outgoing Mailbox 3	=	Incoming Mailbox 3
Outgoing Mailbox 4	=	Incoming Mailbox 4
Incoming Mailbox 1	=	Outgoing Mailbox 1
Incoming Mailbox 2	=	Outgoing Mailbox 2
Incoming Mailbox 3	=	Outgoing Mailbox 3
Incoming Mailbox 4	=	Outgoing Mailbox 4
PCI Mailbox Empty/Full	=	Add-On Mailbox Empty/Full

A write to an outgoing mailbox also writes data into the incoming mailbox on the other interface. It also sets the status bits for the outgoing mailbox and the status bits for the incoming mailbox on the other interface. Reading the incoming mailbox clears all corresponding status bits in the Add-On and PCI mailbox status registers (AMBEF and MBEF).

For example, a PCI write is performed to the PCI outgoing mailbox 2, writing bytes 0 and 1 (BE0# and BE1# asserted). Reading the PCI Mailbox Empty/Full Status Register (MBEF) indicates that bits 4 and 5 are set. These bits indicate that outgoing mailbox 2, bytes 0 and 1 are full. Reading the Add-On Mailbox Empty/Full Status Register (AMBEF) shows that bits 4 and 5 in this register are also set, indicating Add-On incoming mailbox 2, bytes 0 and 1 are full. An Add-On read of incoming mailbox 2, bytes 0 and 1 clears the status bits in both the MBEF and AMBEF status registers.

To reset individual flags in the MBEF and AMBEF registers, the corresponding byte must be read from the incoming mailbox. The PCI and Add-On mailbox status registers, MBEF and AMBEF, are read-only. Mailbox flags may be globally reset from either the PCI interface or the Add-On interface. The PCI Bus Master Control/Status Register (MCSR) and the Add-On General Control/Status Register (AGCSTS) each have a bit to reset all of the mailbox status flags.

Mailbox Interrupts

The designer has the option to generate interrupts to the PCI and Add-On interfaces when specific mailbox events occur. The PCI and Add-On interfaces can each define two conditions where interrupts may be generated. An interrupt can be generated when an incoming mailbox becomes full and/or when an outgoing mailbox becomes empty. A specific byte within a specific mailbox is selected to generate the interrupt. The conditions defined to generate interrupts to the PCI interface do not have to be the same as the conditions defined for the Add-On interface. Interrupts are cleared through software.

For incoming mailbox interrupts, when the specified byte becomes full, an interrupt is generated. The interrupt might be used to indicate command or status information has been provided, and must be read. For PCI incoming mailbox interrupts, the S5933 asserts the PCI interrupt, INTA#. For Add-On incoming mailbox interrupts, the S5933 asserts the Add-On interrupt, IRQ#.

For outgoing mailbox interrupts, when the specified byte becomes empty, an interrupt is generated. The interrupt might be used to indicate that the other interface has received the last information sent and more may be written. For PCI outgoing mailbox interrupts, the S5933 asserts the PCI interrupt, INTA#. For Add-On outgoing mailbox interrupts, the S5933 asserts the Add-On interrupt, IRQ#.

Add-On Outgoing Mailbox 4, Byte 3 Access

PCI incoming mailbox 4, byte 3 (Add-On outgoing mailbox 4, byte 3) does not function exactly like the other mailbox bytes. When an a serial nv memory boot device or no external boot device is used, the S5933 pins EA7:0 are redefined to provide direct external access to Add-On outgoing mailbox 4, byte 3. EA8 is redefined to provide a load clock which may be used to generate a PCI interrupt. The pins are redefined as follows:

Signal Pin	Add-On Outgoing Mailbox
EA0/EMB0	Mailbox 4, bit 24
EA1/EMB1	Mailbox 4, bit 25
EA2/EMB2	Mailbox 4, bit 26
EA3/EMB3	Mailbox 4, bit 27
EA4/EMB4	Mailbox 4, bit 28
EA5/EMB5	Mailbox 4, bit 29
EA6/EMB6	Mailbox 4, bit 30
EA7/EMB7	Mailbox 4, bit 31
EA8/EMBCLK	Mailbox 4, byte 3 load clock

If the S5933 is programmed to generate a PCI interrupt (INTA#), on an Add-On write to outgoing mailbox 4, byte 3, a rising edge on EMBCLK generates a PCI interrupt. The bits EMB7:0 can be read by the PCI bus interface by reading the PCI incoming mailbox 4, byte 3. These bits are useful to indicate various conditions which may have caused the interrupt.

When using the S5933 with a byte-wide boot device, the capability to generate PCI interrupts with Add-On hardware does not exist. In this configuration, PCI incoming mailbox 4, byte 3 (Add-On incoming mailbox 4, byte 3) cannot be used to transfer data from the Add-On - it always returns zeros when read from the PCI bus. This mailbox byte is sacrificed to allow the added functionality provided when a byte-wide boot device is not used.

BUS INTERFACE

The mailboxes appear on the Add-On and PCI bus interfaces as eight operation registers. Four are outgoing mailboxes, four are incoming mailboxes. The mailboxes may be used to generate interrupts to each of the interfaces. The following sections describe the Add-On and PCI bus interfaces for the mailbox registers.

PCI Bus Interface

The mailboxes are only accessible with the S5933 as a PCI target. The mailbox operation registers do not support burst accesses by an initiator. A PCI initiator attempting to burst to the mailbox registers causes the S5933 to respond with a target disconnect with data. PCI writes to full outgoing mailboxes overwrite data currently in that the mailbox. PCI reads from empty incoming mailboxes return the data that was previously contained in the mailbox. Neither of these situations cause a target retry or abort.

PCI incoming and outgoing mailbox interrupts are enabled in the Interrupt Control/Status Register (INTCSR). The mailboxes can generate a PCI interrupt (INTA#) under two conditions (individually enabled). For an incoming mailbox full interrupt, INTA# is asserted on the PCI clock rising edge after the Add-On mailbox write completes. For an outgoing mailbox empty interrupt, INTA# is asserted on the PCI clock rising edge after the Add-On mailbox read completes (the rising edge of RD#). INTA# is deasserted on the next PCI clock rising edge after the PCI access to clear the mailbox interrupt completes (TRDY# deasserted).

Add-On Bus Interface

The Add-On mailbox interface behaves similar to the PCI bus interface. Add-On writes to full outgoing mailboxes overwrite data currently in that mailbox. PCI reads from empty incoming mailboxes return the data that was previously contained in the mailbox.

Add-On incoming and outgoing mailbox interrupts are enabled in the Add-On Interrupt Control/Status Register (AINT). The mailboxes can generate the Add-On IRQ# interrupt under two conditions (individually enabled). For an incoming mailbox full interrupt, IRQ# is asserted one PCI clock period after the PCI mailbox write completes (TRDY# deasserted). For an outgoing mailbox empty interrupt, IRQ# is asserted one PCI clock period after the PCI mailbox read completes (TRDY# deasserted). IRQ# is deasserted immediately when the Add-On clears the mailbox interrupt.

When the S5933 is used with a serial nv memory boot device or no external boot device, the device pins EA8:0 are redefined. EA7:0 become EMB7:0 data inputs and EA8 becomes EMBCLK, a load clock. This configuration allows the Add-On to generate PCI interrupts with a low-to-high transition on EMBCLK. The PCI incoming mailbox interrupt must be enabled and set for mailbox 4, byte3 in the PCI Interrupt Control/Status Register (INTCSR). EMBCLK should begin high and be pulsed low, then high to be recognized. The rising edge of EMBCLK generates the interrupt. The rising edge of EMBCLK also latches in the values on EMB7:0. The S5933 interrupt logic must be cleared (INTA# deasserted) through INTCSR before further EMBCLK interrupts are recognized.

8-Bit and 16-Bit Add-On Interfaces

Some Add-On designs may implement an 8-bit or 16-bit bus interface. The mailboxes do not require a 32-bit Add-On interface. For 8-bit interfaces, the 8-bit data bus may be externally connected to all four bytes of the 32-bit Add-On interface (DQ 31:24, 23:16, 15:8, 7:0 are all connected). The Add-On device reading or writing the mailbox registers may access all mailbox bytes by cycling through the Add-On byte enable inputs. A similar solution applies to 16-bit Add-On buses. This solution works for Add-Ons which always use just 8-bit or just 16-bit accesses.

If the MODE pin is high, indicating a 16-bit Add-On interface, the previous solution may be modified for an 8-bit interface. The difference is that ADR1 must be toggled after the first two accesses to steer the S5933 internal data bus to the upper 16-bits of the mailboxes.

CONFIGURATION

The PCI interface and the Add-On interface each have four incoming mailboxes (IMBx or AIBMx) and four outgoing mailboxes (OMBx or AOMBx) along with a single mailbox status register (MBEF or AMBEF). Outgoing mailboxes are read/write, incoming mailboxes and the mailbox status registers are read-only.

The following sections discuss the registers associated with the mailboxes and accesses required for different modes of mailbox operation.

Mailbox Status

Every byte in each mailbox has a status bit in the Mailbox Empty/Full Status Registers (MBEF and AMBEF). Writing a particular byte into an outgoing mailbox sets the corresponding status bit in both the MBEF and AMBEF registers. A read of a 'full' byte in a mailbox clears the status bit. The MBEF and AMBEF are read-only. Status bits cannot cleared by writes to the status registers.

The S5933 allows the mailbox status bits to be reset through software. The Bus Master Control/Status (MCSR) PCI Operation Register and the Add-On General Control/Status (AGCSTS) Add-On Operation Register each have a bit to reset mailbox status. Writing a '1' to Mailbox Flag Reset bit in the MCSR or the AGCSTS register immediately clears all bits in the both the MBEF and AMBEF registers. Writing a '0' has no effect. The Mailbox Flag Reset bit is write-only.

The flag bits should be monitored when transferring data through the mailboxes. Checking the mailbox status before performing an operation prevents data from being lost or corrupted. The following sequences are suggested for PCI mailbox operations using status polling (interrupts disabled):

Reading a PCI Incoming Mailbox:

1) Check Mailbox Status. Read the mailbox status register to determine if any information has been passed from the Add-On interface.

MBEF	Bits 31:16	If a bit is set, valid data is contained in the corresponding mailbox byte.

2) Read Mailbox(es). Read the mailbox bytes which MBEF indicates are full. This automatically resets the status bits in the MBEF and AMBEF registers.

IMBx	Bits 31:0	Mailbox data.

Writing a PCI Outgoing Mailbox:

1) Check Mailbox Status. Read the mailbox status register to determine if information previously written to the mailbox has been read by the Add-On interface. Writes to full mailbox bytes overwrite data currently in the mailbox (if not already read by the Add-On interface). Repeat until the byte(s) to be written are empty.

MBEF	Bits 15:0	If a bit is set, valid data is contained in the corresponding mailbox byte and has not been read by the Add-On.

2) Write Mailbox(es). Write to the outgoing mailbox byte(s).

OMBx	Bits 31:0	Mailbox data.

Mailbox operations for the Add-On interface are functionally identical. The following sequences are suggested for Add-On mailbox operations using status polling (interrupts disabled):

Reading an Add-On Incoming Mailbox:

1) Check Mailbox Status. Read the mailbox status register to determine if any information has been passed from the PCI interface.

AMBEF	Bits 15:0	If a bit is set, valid data is contained in the corresponding mailbox byte.

2) Read Mailbox(es). Read the mailbox bytes which AMBEF indicates are full. This automatically resets the status bits in the AMBEF and MBEF registers.

AIMBx	Bits 31:0	Mailbox data.

Writing an Add-On Outgoing Mailbox:

1) Check Mailbox Status. Read the mailbox status register to determine if information previously written to the mailbox has been read by the PCI interface. Writes to full mailbox bytes overwrite data currently in the mailbox (if not already read by the PCI interface). Repeat until the byte(s) to be written are empty.

AMBEF	Bits 31:16	If a bit is set, valid data is contained in the corresponding mailbox byte and has not been read by the PCI bus.

2) Write Mailbox(es). Write to the outgoing mailbox byte(s).

AOMBx	Bits 31:0	Mailbox data.

3

Mailbox Interrupts

Although polling status is useful, in some cases, polling requires continuous actions by the processor reading or writing the mailbox. Mailbox interrupt capabilities are provided to avoid much of the processor overhead required by continuously polling status bits.

The Add-On and PCI interface can each generate interrupts on an incoming mailbox condition and/or an outgoing mailbox condition. These can be individually enabled/disabled. A specific byte in one incoming mailbox and one outgoing mailbox is identified to generate the interrupt(s). The tasks required to setup mailbox interrupts are shown below:

Enabling PCI mailbox interrupts:

1) Enable PCI outgoing mailbox interrupts. A specific byte within one of the outgoing mailboxes is identified to assert INTA# when read by the Add-On interface.

INTCSR	Bit 4	Enable outgoing mailbox interrupts
INTCSR	Bits 3:2	Identify mailbox to generate interrupt
INTCSR	Bits 1:0	Identify mailbox byte to generate interrupt

2) Enable PCI incoming mailbox interrupts. A specific byte within one of the incoming mailboxes is identified to assert INTA# when written by the Add-On interface.

INTCSR	Bit 12	Enable incoming mailbox interrupts
INTCSR	Bits 11:10	Identify mailbox to generate interrupt
INTCSR	Bits 9:8	Identify mailbox byte to generate interrupt

Enabling Add-On mailbox interrupts:

1) Enable Add-On outgoing mailbox interrupts. A specific byte within one of the outgoing mailboxes is identified to assert IRQ# when read by the PCI interface.

AINT	Bit 12	Enable outgoing mailbox interrupts
AINT	Bits 11:10	Identify mailbox to generate interrupt
AINT	Bits 9:8	Identify mailbox byte to generate interrupt

2) Enable Add-On incoming mailbox interrupts. A specific byte within one of the incoming mailboxes is identified to assert IRQ# when written by the PCI interface.

AINT	Bit 4	Enable incoming mailbox interrupts
AINT	Bits 3:2	Identify mailbox to generate interrupt
AINT	Bits 1:0	Identify mailbox byte to generate interrupt

With either the Add-On or PCI interface, these two steps can be performed with a single access to the appropriate register. They are shown separately here for clarity.

Once interrupts are enabled, the interrupt service routine must access the mailboxes and clear the interrupt source. A particular application may not require all of the steps shown. For instance, a design may only use incoming mailbox interrupts and not require support for outgoing mailbox interrupts. The interrupt service routine tasks are shown below:

Servicing a PCI mailbox interrupt (INTA#):

1) Identify the interrupt source(s). Multiple interrupt sources are available on the S5933. The interrupt service routine must verify that a mailbox generated the interrupt (and not some other interrupt source).

INTCSR	Bit 16	PCI outgoing mailbox interrupt indicator
INTCSR	Bit 17	PCI incoming mailbox interrupt indicator

2) Check mailbox status. The mailbox status bits indicate which mailbox bytes must be read or written.

MBEF	Bits 31:16	Full PCI incoming mailbox bytes
MBEF	Bits 15:0	Empty PCI outgoing mailbox bytes

3) Access the mailbox. Based on the contents of MBEF, mailboxes are read or written. Reading an incoming mailbox byte clears the corresponding status bit in MBEF.

OMBx	Bits 31:0	PCI outgoing mailboxes
IMBx	Bits 31:0	PCI incoming mailboxes

4) Clear the interrupt source. The PCI INTA# signal is deasserted by clearing the interrupt request. The request is cleared by writing a '1' to the appropriate bit.

INTCSR	Bit 16	Clear PCI outgoing mailbox interrupt
INTCSR	Bit 17	Clear PCI incoming mailbox interrupt

Servicing an Add-On mailbox interrupt (IRQ#):

1) Identify the interrupt source(s). Multiple interrupt sources are available on the S5933. The interrupt service routine must verify that a mailbox generated the interrupt (and not some other interrupt source).

AINT	Bit 16	Add-On incoming mailbox interrupt indicator
AINT	Bit 17	Add-On outgoing mailbox interrupt indicator

2) Check mailbox status. The mailbox status bits indicate which mailbox bytes must be read or written.

AMBEF	Bits 31:16	Empty Add-On outgoing mailbox bytes
AMBEF	Bits 15:0	Full Add-On incoming mailbox bytes

3) Access the mailbox. Based on the contents of AMBEF, mailboxes are read or written. Reading an incoming mailbox byte clears the corresponding status bit in AMBEF.

AIMBx	Bits 31:0	Add-On incoming mailboxes
AOMBx	Bits 31:0	Add-On outgoing mailboxes

4) Clear the interrupt source. The Add-On IRQ# signal is deasserted by clearing the interrupt request. The request is cleared by writing a '1' to the appropriate bit.

AINT	Bit 16	Clear Add-On incoming mailbox interrupt
AINT	Bit 17	Clear Add-On outgoing mailbox interrupt

In both cases, step 3 involves accessing the mailbox. To allow the incoming mailbox interrupt logic to be cleared, the mailbox status bit must also be cleared. Reading an incoming mailbox clears the status bits. Another option for clearing the status bits is to use the Mailbox Flag Reset bit in the MCSR and AGCSTS registers, but this clears all status bits, not just for a single mailbox or mailbox byte. For outgoing mailbox interrupts, the read of a mailbox register is what generated the interrupt; this ensures the status bits are already clear.

PAGE(S) INTENTIONALLY BLANK

FIFO OVERVIEW

The S5933 has two internal FIFOs. One FIFO is for PCI bus to Add-On bus, the other FIFO is for Add-On bus to PCI bus transfers. Each of these has eight 32-bit registers. The FIFOs are both addressed through a single PCI/Add-On Operation Register offset, but which internal FIFO is accessed is determined by whether the access is a read or write.

The FIFO may be either a PCI target or a PCI initiator. As a target, the FIFO allows a PCI bus master to access Add-On data. The FIFO also allows the S5933 to become a PCI initiator. Read and write address registers and transfer count registers allow the S5933 to perform DMA transfers across the PCI bus. The FIFO may act as initiator and a target at different times in the same application.

The FIFO can be configured to support various Add-On bus configurations. FIFO status and control signals allow simple cascading into an external FIFO, the Add-On bus can be 8-, 16-, or 32-bits wide, and data endian conversion is optional to support any type of Add-On CPU. PCI and Add-On interrupt capabilities are available to support bus mastering through the FIFO.

FUNCTIONAL DESCRIPTION

The S5933 FIFO interface allows a high degree of functionality and flexibility. Different FIFO management schemes, endian conversion schemes, and advance conditions allow for a wide variety of Add-On interfaces. Applications may implement the FIFO as either a PCI target or program it to enable the S5933 to be a PCI initiator (bus master). The following sections describe, on a functional level, the capabilities of the S5933 FIFO interface.

FIFO Buffer Management and Endian Conversion

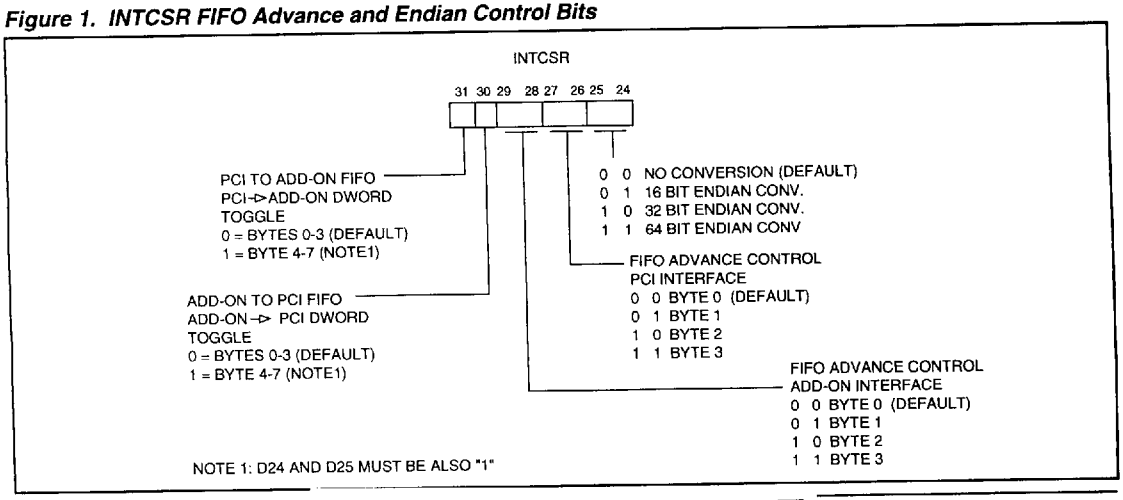
The S5933 provides a high degree of flexibility for controlling the data flow through the FIFO. Each FIFO (PCI to Add-On and Add-On to PCI) has a specific FIFO advance condition. For FIFO writes, the byte which signifies a location is full is configurable. For FIFO reads, the byte which signifies a location is empty is configurable. This ability is useful for transferring data through the FIFO with Add-Ons which are not 32-bits wide. Endian conversion may also be performed on data passing through the FIFO.

FIFO Advance Conditions

The specific byte lane used to advance the FIFO, when accessed, is determined individually for each FIFO interface (PCI and Add-On). The control bits to set the advance condition are D29:26 of the Interrupt Control/Status Register (INTCSR) in the PCI Operation Registers (Figure 1). The default FIFO advance condition is set to byte 0. With the default setting, a write to the FIFO with BE0# asserted indicates that the FIFO location is now full, advancing the FIFO pointer to the next location. BE0# does not have to be the only byte enable asserted. Note, the FIFO advance condition may be different for the PCI to Add-On FIFO and the Add-On to PCI FIFO directions.

3

Figure 1. INTCSR FIFO Advance and Endian Control Bits



The configurable FIFO advance condition may be used to transfer data to and from Add-On interfaces which are not 32-bits wide. For a 16-bit Add-On bus, the Add-On to PCI FIFO advance condition can be set to byte 2. This allows a 16-bit write to the lower 16-bits of the FIFO register (bytes 0 and 1) and a second write to the upper 16-bits of the FIFO register (bytes 2 and 3). The FIFO does not advance until the second access. This allows the Add-On to operate with 16-bit data, while the PCI bus maintains a 32-bit data path.

Notes:

1. During operation, the INTCSR FIFO advance condition bits (D29:26) should only be changed when the FIFO is empty and is idle on both the Add-On and PCI interfaces.

Endian Conversion

Bits D31:30 and D25:24 of the INTCSR PCI Operation Register control endian conversion operations for the FIFO (Figure 1). When endian conversion is performed, it affects data passing in either direction through the FIFO interface. Figures 2a and 2b show 16-bit and 32-bit endian conversion. It is important to note that endian conversion is performed on data BEFORE it enters the FIFO. This affects the FIFO advance condition. Example: the FIFO is configured to perform 32-bit endian conversion on data, and the FIFO advance condition is set to byte 0. Byte 3 is written into the FIFO (BE3# asserted). After the endian conversion, byte 3 becomes byte 0, and the FIFO advances. This behavior must be considered when not performing full 32-bit accesses to the FIFO.

Notes:

1. During operation, the INTCSR FIFO endian conversion bits (D25:24) and 64-bit access bits (D31:30) should only be changed when the FIFO is empty and is idle on both the Add-On and PCI interfaces.

Figure 2a. 16-bit Endian Conversion

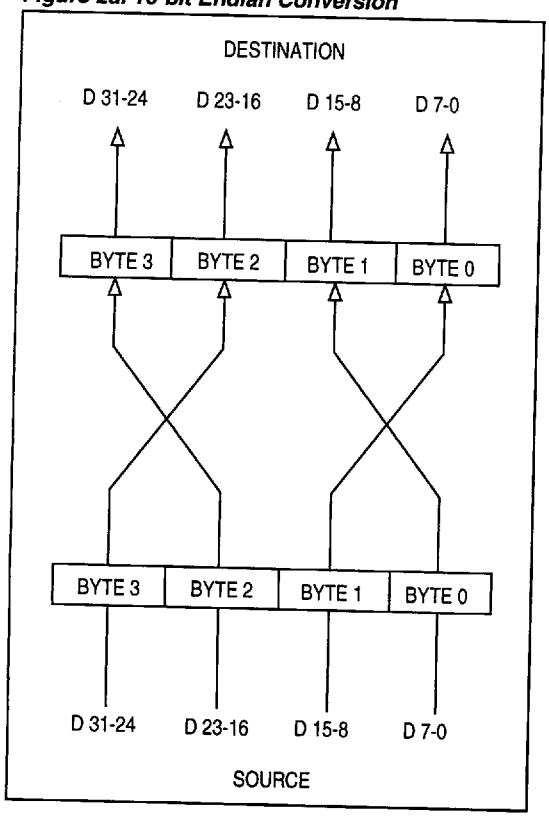
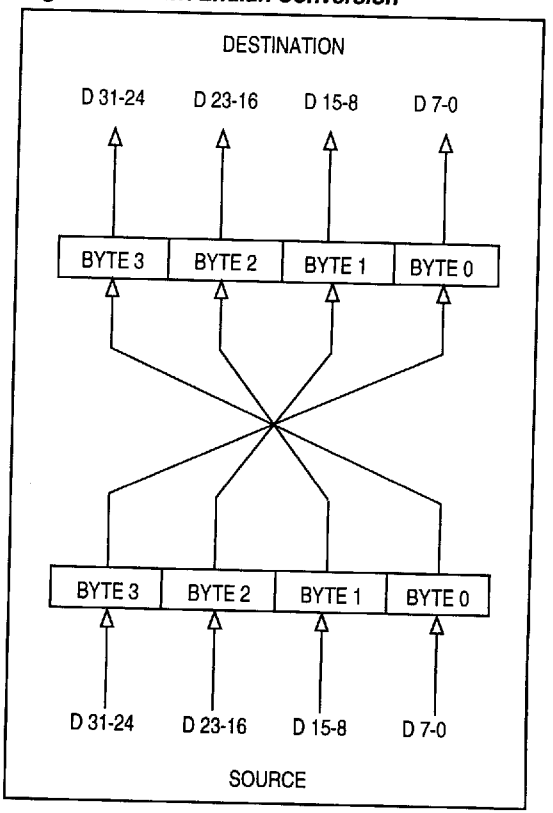


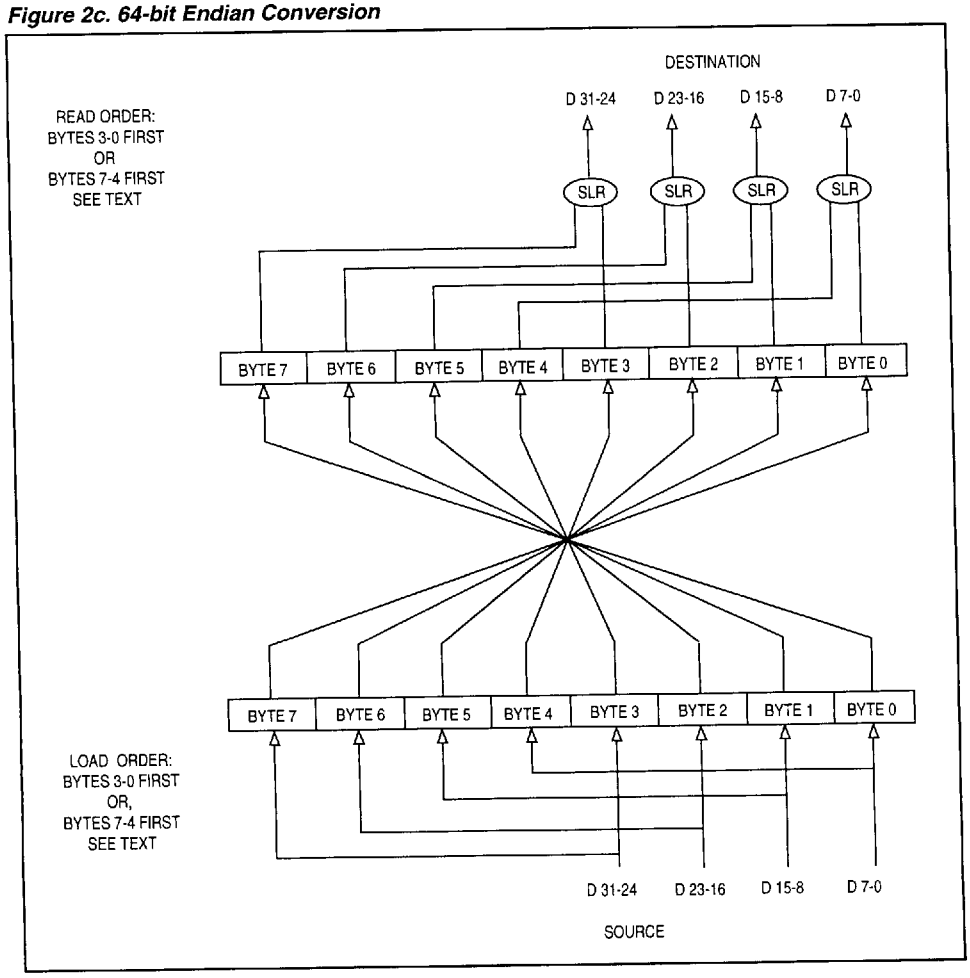
Figure 2b. 32-bit Endian Conversion



64-Bit Endian Conversion

Because the S5933 interfaces to a 32-bit PCI bus, special operation is required to handle 64-bit data endian conversion. Figure 2c shows 64-bit endian conversion. The S5933 must know whether the lower 32-bits enter the FIFO first or the upper 32-bits enter the FIFO first. INTCSR D31:30 identify which method is used by the application. These bits toggle after each 32-bit operation to indicate if half or all of a 64-bit data operation has been completed. The initial state of these bits establishes the loading and emptying order for 64-bit data during operation.

Figure 2c. 64-bit Endian Conversion



Add-On FIFO Status Indicators

The Add-On interface implements FIFO status pins to indicate the full and empty conditions of the PCI to Add-On and Add-On to PCI FIFOs. These may be used by the Add-On to allow data transfers between the FIFO and memory, a peripheral, or even a cascaded external FIFO. The RDEMPTY and WRFULL status outputs are always available to the Add-On. Additional status signals are multiplexed with the byte-wide, non-volatile memory interface pins. If the S5933 is configured for Add-On initiated bus mastering, these status signals also become available to the Add-On. FIFO status is also indicated by bits in the Add-On General Control/Status and Bus Master Control/Status Registers. The table below lists all FIFO status outputs and their functions.

Signal	Function
RDEMPTY	Indicates empty condition of the PCI to Add-On FIFO
WRFULL	Indicates full condition of the Add-On to PCI FIFO
FRF	Indicates full condition of the PCI to Add-On FIFO [1]
FWE	Indicates the empty condition of the Add-On to PCI FIFO [1]

1. These signals are only available when a serial non-volatile memory is used and the device is configured for Add-On initiated bus mastering.

Add-On FIFO Control Signals

The Add-On interface implements FIFO control pins to manipulate the S5933 FIFOs. These may be used by Add-On to control data transfer between the FIFO and memory, a peripheral, or even a cascaded external FIFO. The RDFIFO# and WRFIFO# inputs are always available. These pins allow direct access to the FIFO without generating a standard Add-On register access using RD#, WR#, SELECT#, address pins and the byte enables.

Additional control signals are multiplexed with the byte-wide, non-volatile memory interface pins. If a serial non-volatile memory is used and the S5933 is configured for Add-On initiated bus mastering, these control signals also become available. For PCI initiated bus mastering, AMREN, AMWEN, FRC#, and FWC# functionality is always available through bits in the Bus Master Control/Status and Add-On General Control/Status Registers. The FIFO control inputs are listed below.

Signal	Function
RDFIFO#	Reads data from the PCI to Add-On FIFO
WRFIFO#	Writes data into the Add-On to PCI FIFO
FRC#	Reset PCI to Add-On FIFO pointers and status indicators [1]
FWC#	Reset Add-On to PCI FIFO pointers and status indicators [1]
AMREN	Enable bus mastering for Add-On initiated PCI reads [1]
AMWEN	Enable bus mastering for Add-On initiated PCI writes [1]

1. These signals are only available when a serial non-volatile memory is used and the S5933 is configured for Add-On initiated bus mastering.

PCI Bus Mastering with the FIFO

The S5933 may initiate PCI bus cycles through the FIFO interface. The S5933 allows blocks of data to be transferred to and from the Add-On by specifying a source/destination address on the PCI bus and a transfer byte count. This DMA capability allows data to be transferred across the PCI bus without host CPU intervention.

Initiating a bus master transfer requires programming the appropriate address registers and transfer byte counts. This can be done from either the PCI interface or the Add-On interface. Initiating bus master transfers from the add-on is advantageous because the host CPU does not have to intervene for the S5933 to become a PCI Initiator. At the end of a transfer the S5933 may generate an interrupt to either the PCI bus (for PCI initiated transfers) or Add-On interface (for Add-On initiated transfers).

Add-On Initiated Bus Mastering

If bit 7 in location 45h of an external serial non-volatile memory is zero, the Master Read Address Register (MRAR), Master Write Address Register (MWAR), Master Read Transfer Count (MRTC), and Master Write Transfer Count (MWTC) are accessible only from the Add-On interface. Add-On initiated bus mastering is not possible when a byte-wide boot device is used due to shared device pins. When configured for Add-On initiated bus mastering, the S5933 transfers data until the transfer count reaches zero, or it may be configured to ignore the transfer count.

For bus master transfers initiated by the Add-On interface, some applications may not know the size of the data block to be transferred. To avoid constantly updating the transfer count register, the transfer count may be disabled. Bit 28 in the Add-On General Control/Status Register (AGCSTS) performs this function. Disabling the transfer count also disables the interrupt capabilities. Regardless of whether Add-On transfer count is enabled or disabled, the Add-On Master Read Enable (AMREN) and Add-On Master Write Enable (AMWEN) inputs control when the S5933 asserts or deasserts its request to the PCI bus. When Add-On transfer count is enabled, the S5933 will only request the bus when both the transfer count (read or write) is not zero and the appropriate enable line (AMREN or AMWEN) is active. For Add-On initiated bus mastering, AMWEN and AMREN override the read and write bus mastering enable bits in the Bus Master Control/Status Register (MCSR).

PCI Initiated Bus Mastering

If bit 7 in location 45h of the external non-volatile memory is one, the Master Read Address Register (MRAR), Master Write Address Register (MWAR), Master Read Transfer Count (MRTC), and Master Write Transfer Count (MWTC) are accessible only from the PCI bus interface. In this configuration, the S5933 transfers data until the transfer count reaches zero. The transfer count cannot be disabled for PCI initiated bus mastering. If no external nv memory boot device is used, the S5933 defaults to PCI initiated bus mastering.

Address and Transfer Count Registers

The S5933 has two sets of registers used for bus master transfers. There are two operation registers for bus master read operations and two operation registers for bus master write operations. One operation register is for the transfer address (MWAR and MRAR). The other operation register is for the transfer byte count (MWTC and MRTC).

The address registers are written with the first address of the transfer before bus mastering is enabled. Once a transfer begins, this register is automatically updated to reflect the address of the current transfer. If a PCI target disconnects from an S5933 initiated cycle, the transfer is retried starting from the current address in the register. If bus grant (GNT#) is removed or bus mastering is disabled (using AMREN or AMWEN), the value in the address register reflects the next address to be accessed. Transfers must begin on DWORD boundaries.

The transfer count registers contain the number of bytes to be transferred. The transfer count may be written before or after bus mastering is enabled. If bus mastering is enabled, no transfer occurs until the transfer count is programmed with a non-zero value. Once a transfer begins, this register is automatically updated to reflect the number of bytes remaining to be transferred. If the transfer count registers are disabled (for Add-On initiated bus mastering), transfers begin as soon as bus mastering is enabled.

Although transfers must begin on DWORD boundaries, transfer counts do not have to be multiples of four bytes. For example, if the write transfer count (MWTC) register is programmed with a value of 10 (decimal), the S5933 performs two DWORD writes and a third write with only BE0# and BE1# asserted.

Bus Mastering FIFO Management Schemes

The S5933 provides flexibility in how the FIFO is managed for bus mastering. The FIFO management scheme determines when the S5933 requests the bus to initiate PCI bus cycles. The management scheme is configurable for the PCI to Add-On and Add-On to PCI FIFO (and may be different for each). Bus mastering must be enabled for the management scheme to apply (via the enable bits or AMREN/AMWEN).

For the PCI to Add-On FIFO, there are two management options. The PCI to Add-On FIFO management option is programmed through the Bus Master Control/Status Register (MCSR). The FIFO can be programmed to request the bus when any DWORD location is empty or only when four or more locations are empty. After the S5933 is granted control of the PCI bus, the management scheme does not apply. The device continues to read as long as there is an open FIFO location. When the PCI to Add-On FIFO is full or bus mastering is disabled, the PCI bus request is removed by the S5933.

For the Add-On to PCI FIFO, there are two management options. The Add-On to PCI FIFO management option is programmed through the Bus Master Control/Status Register (MCSR). The FIFO can be programmed to request the bus when any DWORD location is full or only when four or more locations are full. After the S5933 is granted control of the PCI bus, the management scheme does not apply. The device continues to write as long as there is data in the FIFO. When the Add-On to PCI FIFO is empty or bus mastering is disabled, the PCI bus request is removed by the S5933.

There are two special cases for the Add-On to PCI FIFO management scheme. The first case is when the FIFO is programmed to request the PCI bus only when four or more locations are full, but the transfer count is less than 16 bytes. In this situation, the FIFO ignores the management scheme and finishes transferring the data. The second case is when the S5933 is configured for Add-On initiated bus mastering with transfer counts disabled. In this situation, the FIFO management scheme must be set to request the PCI bus when one or more locations are full. AMREN and AMWEN may be used to implement a specific FIFO management scheme.

FIFO Bus Master Cycle Priority

In many applications, the FIFO is used as a PCI initiator performing both PCI reads and writes. This requires a priority scheme be implemented. What happens if the FIFO condition for initiating a PCI read and a PCI write are both met?

Bits D12 and D8 in the Bus Master Control/Status Register (MCSR) control the read and write cycle priority, respectively. If these bits are both set or both clear, priority alternates, beginning with a read cycle. If the read priority is set and the write priority is clear, read cycles take priority. If the write priority is set and the read priority is clear, write cycles take priority. Priority arbitration is only done when neither FIFO has control of the PCI bus (the PCI to Add-On FIFO would never interrupt an Add-On to PCI FIFO transfer).

FIFO Generated Bus Master Interrupts

Interrupts may be generated under certain conditions from the FIFO. If PCI initiated bus mastering is used, INTA# is generated to the PCI interface. If Add-On initiated bus mastering is used, IRQ# is generated to the Add-On interface. Interrupts may be disabled.

FIFO Interrupts may be generated from one or more of the following during bus mastering: read transfer count reaches zero, write transfer count reaches zero, or an error occurs during bus mastering. Error conditions include a target or master abort on the PCI bus. Interrupts on PCI error conditions are only enabled if one or both of the transfer count interrupts are enabled.

The Add-On Interrupt Control/Status Register (AINT) or the Interrupt Control Status Register (INTCSR) indicates the interrupt source. The interrupt service routine may read these registers to determine what action is required. As mailboxes are also capable of generating interrupts, this must also be considered in the service routine. Interrupts are also cleared through these registers.

BUS INTERFACE

The S5933 FIFO may be accessed from the Add-On interface or the PCI interface. Add-On FIFO control and status signals allow a simple interface to the FIFO with either an Add-On CPU or programmable logic. The following section describes the PCI and Add-On interface behavior and hardware interface.

FIFO PCI Interface (Target Mode)

The S5933 FIFO may act as a standard PCI target. FIFO empty/full status may be determined by the PCI initiator by reading the status bits in the PCI Bus Master Control/Status Register (MCSR).

The FIFO occupies a single 32-bit register location within the PCI Operation Registers. **A PCI initiator may not perform burst accesses on the FIFO.** Each data phase of a burst causes the PCI initiator to increment its address counter (even though only the first address is driven at the beginning of the burst). The initiator keeps track of the current address in case a disconnect occurs. This allows the initiator to continue the burst from where the disconnect occurred. If the S5933 FIFO initiated a disconnect during a PCI burst to the FIFO register, the burst would be resumed at an address other than the FIFO location (because the initiator address counter has incremented). The S5933 always signals a disconnect if a burst to any PCI Operation Register is attempted.

Because the PCI to Add-On FIFO and the Add-On to PCI FIFO occupy a single location within the PCI and Add-On Operation Registers, which FIFO is accessed is determined by whether the access is a read or write. This means that once data is written into the FIFO, the value written cannot be read back.

For PCI reads from the Add-On to PCI FIFO, the S5933 asserts TRDY# and completes the PCI cycle (Figure 3). If the PCI bus attempts to read an empty FIFO, the S5933 immediately issues a disconnect with retry (Figure 4). The Add-On to PCI FIFO status indicators change one PCI clock after a PCI read.

For PCI writes to the PCI to Add-On, the S5933 asserts TRDY# and completes the PCI cycle (Figure 5). If the PCI bus attempts to write a full FIFO, the S5933 immediately issues a disconnect with retry (Figure 6). The PCI to Add-On FIFO status indicators change one PCI clock after a PCI write.

FIFO PCI Interface (Initiator Mode)

The S5933 can act as an initiator on the PCI bus. This allows the device to gain control of the PCI bus to transfer data to or from the FIFO. Internal address and transfer count registers control the number of PCI transfers and the locations of the transfers. The following paragraphs assume the proper registers and bits are programmed to enable bus mastering .

PCI read and write transfers from the S5933 are very similar. The FIFO management scheme determines when the S5933 asserts its PCI bus request (REQ#). When bus grant (GNT#) is returned, the device begins running PCI cycles. Once the S5933 controls the bus, the FIFO management scheme is not important. It only determines when PCI bus control is initially requested. PCI bus reads and writes are always performed as bursts by the S5933, if possible.

Figure 3. PCI Read from a Full S5933 FIFO

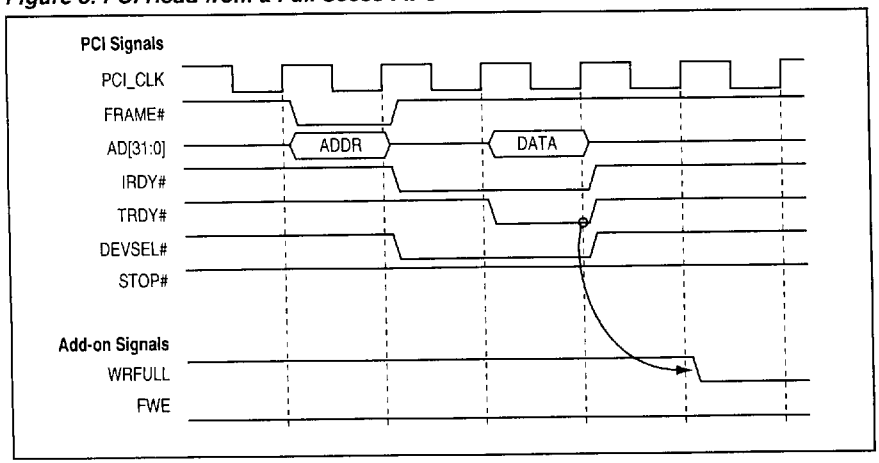


Figure 4. PCI Read from an Empty S5933 FIFO (Target Disconnect)

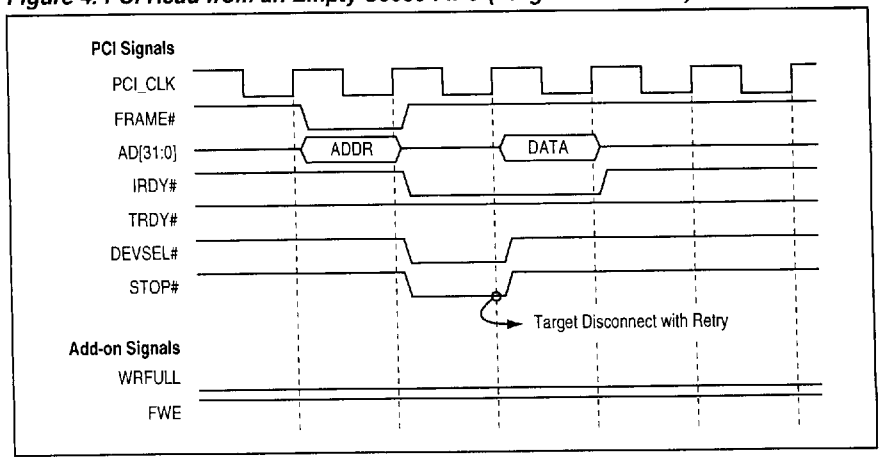


Figure 5. PCI Write to an Empty S5933 FIFO

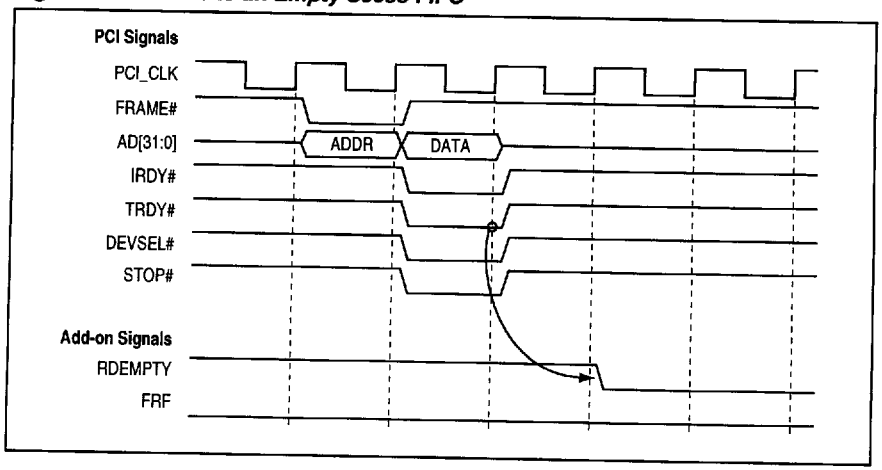
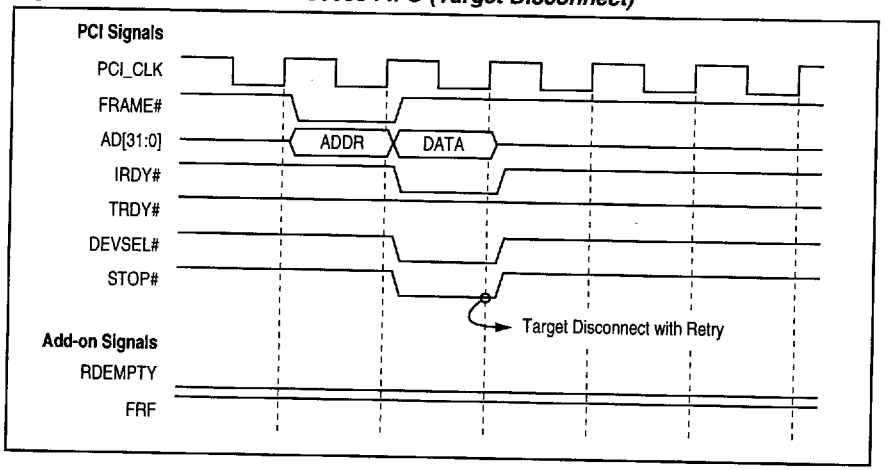


Figure 6. PCI Write to a Full S5933 FIFO (Target Disconnect)



FIFO PCI Bus Master Reads

For PCI read transfers (filling the PCI to Add-On FIFO), read cycles are performed until one of the following occurs:

- Bus Master Read Transfer Count Register (MRTC), if used, reaches zero
- The PCI to Add-On FIFO is full
- GNT# is removed by the PCI bus arbiter
- AMREN is deasserted

If the transfer count is not zero, GNT# remains asserted, and AMREN is asserted, the FIFO continues to read data from the PCI bus until there are no empty locations in the PCI to Add-On FIFO. If the Add-On can empty the FIFO as quickly as it can be filled from the PCI bus, very long bursts are possible. The S5933 deasserts REQ# when it completes the access to fill the last location in the FIFO. Once REQ# is deasserted, it will not be reasserted until the FIFO management condition is met.

FIFO PCI Bus Master Writes

For PCI write transfers (emptying the Add-On to PCI FIFO), write cycles are performed until one of the following occurs:

- Bus Master Write Transfer Count Register (MWTC), if used, reaches zero
- The Add-On to PCI FIFO is empty
- GNT# is removed by the PCI bus arbiter
- AMWEN is deasserted

If the transfer count is not zero, GNT# remains asserted, and AMWEN is asserted, The FIFO continues to write data to the PCI bus until there are is no data in the Add-On to PCI FIFO. If the Add-On can fill the FIFO as quickly as it can be emptied to the PCI bus, very long bursts are possible. The S5933 deasserts REQ# when it completes the access to transfer the last data in the FIFO. Once REQ# is deasserted, it will not be reasserted until the FIFO management condition is met.

Add-On Bus Interface

The FIFO register may be accessed in two ways from the Add-On interface. It can be accessed through normal register accesses or directly with the RDFIFO# and WRFIFO# inputs. In addition, the FIFO register can be accessed with synchronous or asynchronous to BPCLK, depending on the S5933 configuration. The Add-On interface also supports datapaths which are not 32-bits. The method used to access the FIFO from the Add-On interface is independent of whether the FIFO is a PCI PCI target or a PCI initiator.

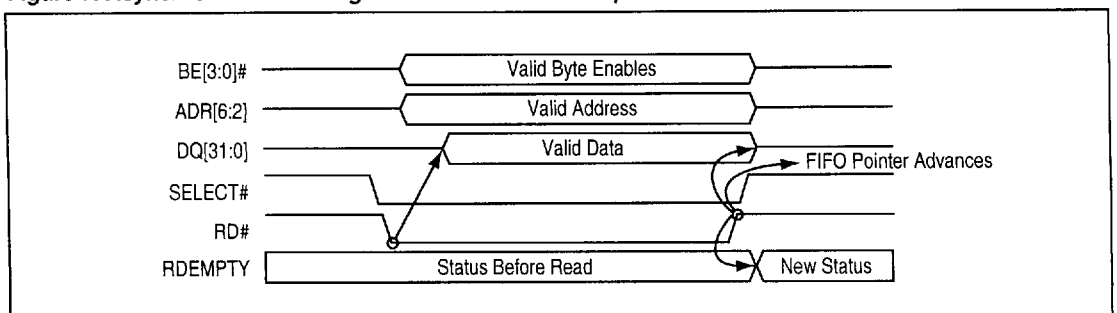
Add-On FIFO Register Accesses

The FIFO may be accessed from the Add-On interface through the Add-On FIFO Port Register (AFIFO) read or write. This is offset 20h in the Add-On Operation Registers. Depending on the device configuration, this register can be accessed either synchronous BPCLK or asynchronous to BPCLK. To access the FIFO as a normal Add-On Operation Register, ADR[6:2], BE[3:0]#, SELECT#, and RD# or WR# are required. The major differences between synchronous and asynchronous modes are when the FIFO pointers advance and the ability to perform burst accesses. The following examples are shown for Add-On FIFO reads.

Figure 7 shows an asynchronous FIFO register read. SELECT# must meet setup and hold times relative to the rising edge of RD#. RD# and SELECT# both asserted enables the DQ outputs, and the first data location in the FIFO is driven onto the bus. The FIFO address and the byte enables must be valid before valid data is driven onto the DQ bus. Data remains valid as long as the address, byte enables, SELECT# and RD# are asserted. Deasserting RD# or SELECT# causes the data bus to float. The rising edge of RD# causes the FIFO pointer to advance. The status outputs are updated to reflect the FIFO condition after it advances.

3

Figure 7. Asynchronous FIFO Register Read Access Example



When the last location in the PCI to Add-On FIFO is read by the Add-On, the FIFO pointer does not change. If another read is performed before more data enters the FIFO, the previous data is driven. When a write to a full Add-On to PCI FIFO is attempted, nothing happens. No FIFO data is overwritten and the FIFO pointers are not changed. This behavior is the same whether the FIFO is accessed using the direct access inputs or normal Operation Register accesses.

Figure 8 shows a synchronous FIFO register burst access. SELECT# must meet setup and hold times relative to the rising edge of BPCLK. RD# and SELECT# both asserted enables the DQ outputs, and the first data location (data 0) in the FIFO is driven on to the bus. The FIFO address and the byte enables must be valid before valid data is driven onto the DQ bus. Data 0 remains valid until the next rising edge of BPCLK. The rising edge of BPCLK causes the FIFO pointer to advance to the next location (data 1). The next rising edge of BPCLK also advances the FIFO pointer to the next location (data 2). The status outputs reflect the FIFO condition after it advances, and are updated off of the rising edge of BPCLK. When RD# or SELECT# is deasserted, the DQ bus floats. The next time a valid FIFO access occurs and RD# and SELECT# are asserted, data 2 is presented on the DQ bus (as there was no BPCLK edge to advance the FIFO).

Add-On FIFO Direct Access Mode

Instead of generating an address, byte enables, SELECT# and a RD# or WR# strobe for every FIFO access, the S5933 allows a simple, direct access mode. Using RDFIFO# and WRFIFO# is functionally identical to performing a standard AFIFO Port Register access, but requires less logic to implement. Accesses to the FIFO register using the direct access signals are always 32-bits wide. The only exception to this is when the MODE pin is configured for 16-bit operation. In this situation, all accesses are 16-bits wide. The RD# and WR# inputs must be inactive when RDFIFO# or WRFIFO# is active. The ADR[6:2] and BE[3:0]# inputs are ignored.

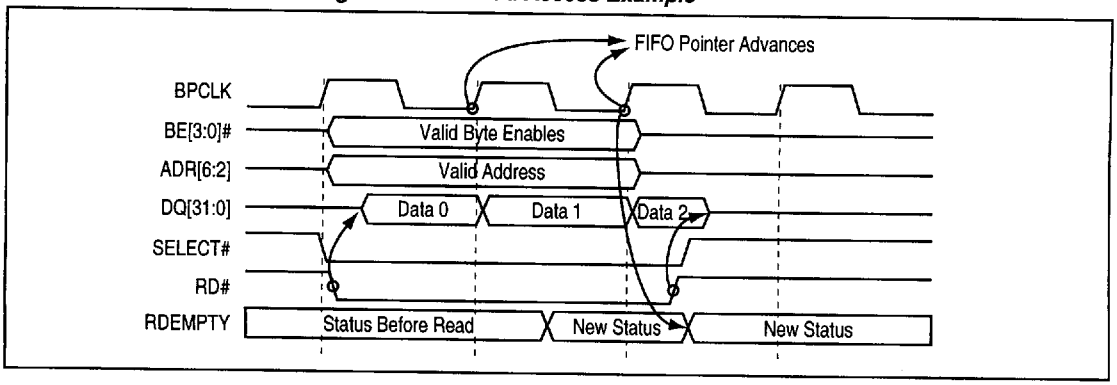
Depending on the device configuration, RDFIFO# and WRFIFO# can act as clocks for data or enables with BPCLK acting as the clock. A Synchronous interface allows higher data rates, and an asynchronous interface is better for slow Add-On logic which may require wait states. The major difference between the synchronous and asynchronous modes is when the FIFO advances.

Figure 9 shows an asynchronous FIFO register direct access using RDFIFO#. The first location in the FIFO is driven onto the bus when RDFIFO# is asserted. Data remains valid as long as RDFIFO# is asserted. The rising edge of RDFIFO# causes the data bus to float and acts as the clock causing the FIFO pointer to advance. The status outputs reflect the FIFO condition after it advances.

Figure 10 shows a synchronous FIFO register direct burst access using RDFIFO#. RDFIFO# acts as an enable and the first data location (data 0) in the FIFO is driven on to the bus when RDFIFO# is asserted. Data 0 remains valid until the next rising edge of BPCLK. The rising edge of BPCLK causes the FIFO pointer to advance to the next location (data 1). The next rising edge of BPCLK advances the FIFO pointer to the next location (data 2). The status outputs reflect the FIFO condition after it advances, and are updated off of the rising edge of BPCLK. When RDFIFO# is deasserted, the DQ bus floats. The next time RDFIFO# is asserted, data 2 is presented on the DQ bus (as there was no BPCLK edge to advance the FIFO).

A synchronous FIFO interface has the advantage of allowing data to be accessed more quickly (in bursts) by the Add-On. As a target, if a full S5933 FIFO is

Figure 8. Synchronous FIFO Register Burst Read Access Example



written (or an empty FIFO is read) by a PCI initiator, the S5933 requests a retry. The faster the Add-On interface can empty (or fill) the FIFO, the less often retries occur. With the S5933 as a PCI initiator, a similar situation occurs. Not emptying or filling the FIFO quickly enough results in the S5933 giving up control of the PCI bus. Higher PCI bus data transfer rates are possible through the FIFO with a synchronous interface.

Additional Status/Control Signals for Add-On Initiated Bus Mastering

If a serial non-volatile memory is used to configure the S5933, and the device is configured for Add-On initiated bus mastering, two additional FIFO status signals and four additional control signals are available to the Add-On interface. The FRF and FWE outputs provide additional FIFO status information. Inputs FRC#, FWC#, AMREN, and AMWEN provide additional FIFO control. Applications may use these signals to monitor/control FIFO flags and PCI bus requests. These new signals are some of the lines that were used for byte-wide nvram interface, but now are reconfigured. The reconfigured lines are as follows:

Outputs:

E_ADDR (15) FRF
FIFO Read Full: Indicates that the PCI to Add-On FIFO is full.

E_ADDR (14) FWE
FIFO Write Empty: Indicates that the Add-On to PCI FIFO is empty.

Inputs:

EQ (7) AMWEN
Add-On bus Mastering Write ENable: This input is driven high to enable bus master writes.

EQ (6) AMREN
Add-On bus Mastering Read ENable: This input is driven high to enable bus master reads.

EQ (5) FRC#
FIFO Read Clear: This line is driven low to clear the PCI to Add-On FIFO.

EQ (4) FWC#
FIFO Write Clear: This line is driven low to clear the Add-On to PCI FIFO.

FRF (PCI to Add-On FIFO full) and FWE (Add-On to PCI FIFO empty) supplement the RDEMPTY and WRFULL status indicators. These additional status outputs provide additional FIFO status information for Add-On FIFO control logic.

3

Figure 9. Asynchronous FIFO Register RDFIFO# Access Example

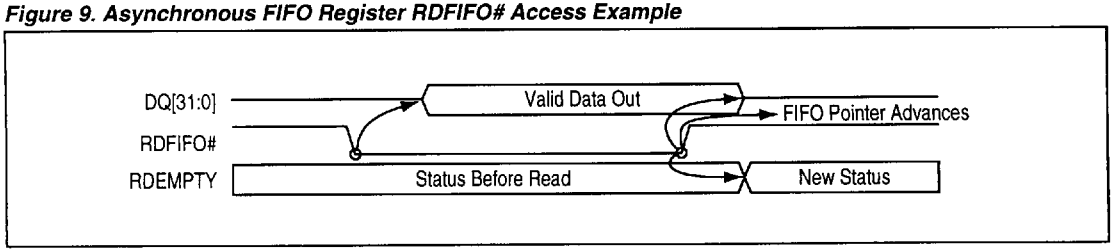
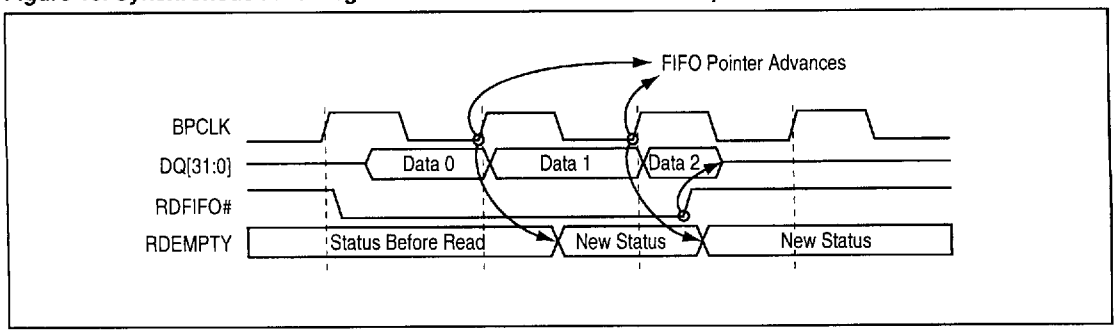


Figure 10. Synchronous FIFO Register Burst RDFIFO# Access Example



The FRC# and FWC# inputs allow Add-On logic to reset the PCI to Add-On or Add-On to PCI FIFO flags. The FIFO flags can always be reset with software through the Add-On General Control/Status Register (AGCSTS) or the Bus Master Control/Status Register (MCSR), but these hardware inputs are useful for designs which do no implement a CPU on the Add-On card. Asserting the FRC# input resets the PCI to Add-On FIFO. Asserting the FWC# input resets the Add-On to PCI FIFO.

The AMREN and AMWEN inputs allow Add-On logic to individually enable and disable bus mastering for the PCI to Add-On and Add-On to PCI FIFO. These inputs override the Bus Master Control/Status Register (MCSR) bus master enable bits. The S5933 may request the PCI bus for the PCI to Add-On FIFO when AMREN is asserted and may request the PCI bus for the Add-On to PCI FIFO when AMWEN is asserted. If AMREN or AMWEN is deasserted, the S5933 removes its PCI bus request and gives up control of the bus.

AMREN and AMWEN are useful for Add-Ons with external FIFOs cascaded into the S5933. For PCI bus master write operations, the entire S5933 Add-On to PCI FIFO and the external FIFO may be filled before enabling bus mastering, providing a single long burst write rather than numerous short bursts.

In some applications, the amount of data to be transferred is not known. During read operations, the S5933, attempting to fill its PCI to Add-On FIFO, may access up to eight memory locations beyond what is required by the Add-On before it stops. In this situation, AMREN can be deasserted to disable PCI reads, and then FRC# can be asserted to flush the unwanted data from the FIFO.

FIFO Generated Add-On Interrupts

For Add-On initiated bus mastering, the S5933 may be configured to generate interrupts to the Add-On interface for the following situations:

- Read transfer count reaches zero
- Write transfer count reaches zero
- An error occurred during the bus master transaction

The interrupt is posted to the Add-On interface with the IRQ# output. A high-to-low transition on this output indicates an interrupt condition. Because there is a single interrupt output and multiple interrupt conditions, the Add-On Interrupt Control/Status Register (AINT) must be read to determine the interrupt source. This register is also used to clear the interrupt, returning IRQ# to its high state. If mailbox interrupts are also used, this must be considered in the interrupt service routine.

8-Bit and 16-Bit FIFO Add-On Interfaces

The S5933 FIFO may also be used to transfer data between the PCI bus and 8-bit or 16-bit Add-On interfaces. This can be done using FIFO advance conditions or the S5933 MODE input pin.

The FIFO may be used as an 8-bit or 16-bit wide FIFO. To use the FIFO as an 8-bit interface, the advance condition should be set for byte 0 (no data is transferred in the upper 3 bytes). To use the FIFO as a 16-bit interface, the advance condition should be set for byte 1 (no data is transferred in the upper 2 bytes). This allows a simple Add-On bus interface, but it has the disadvantage of not efficiently utilizing the PCI bus bandwidth because the host is forced to perform 8-bit or 16-bit accesses to the FIFO on the PCI bus. This is the only way to communicate with an 8-bit Add-On through the FIFO without additional logic to steer byte lanes on the Add-On data bus. Pass-Thru mode is more suited to 8-bit Add-On interfaces.

Implementing a 16-bit wide FIFO is a reasonable solution, but to avoid wasting PCI bus bandwidth, the best method is to allow the PCI bus and the FIFO to operate with 32-bit data. The S5933 can assemble or disassemble 32-bit quantities for the Add-On interface. This is possible through the MODE pin. When MODE is low, the Add-On data bus is 32-bits. When MODE is high, the Add-On data bus is 16-bits. When MODE is configured for 16-bit operation, BE3# becomes ADR1.

With the FIFO direct access signals (RDFIFO# and WRFIFO#), the MODE pin must reflect the actual Add-On data bus width. With MODE = 16-bits, the S5933 automatically takes two consecutive, 16-bit Add-On writes to the FIFO and assembles a 32-bit value. FIFO reads operate in the same manner. Two consecutive Add-On reads empty the 32-bit FIFO register. The 16-bit data bus is internally steered to the lower and upper words of the 32-bit FIFO register.

One consideration needs to be taken when using the FIFO direct access signals and letting the S5933 do byte lane steering internally. The default condition used to advance the FIFO is byte 0. This must be changed to byte 2 or 3. When MODE is configured for a 16-bit Add-On bus, the first 16-bit cycle to the FIFO always accesses the low 16-bits. If the FIFO advance condition is left at byte 0, the FIFO advances after the first 16-bit cycle and the data in the upper 16-bits is directed to the next FIFO location, shifting the data.

Some applications hold the RDFIFO# and WRFIFO# inputs active for a synchronous interface. In 16-bit mode, designs must avoid writing to a full FIFO. The data for the write is lost, but the internal mechanism to direct the 16-bit external data bus to the upper 16-bits of the FIFO register is triggered. This creates a situation where the FIFO is out of step. The next 16-bit FIFO write is directed to the upper 16-bits of the FIFO, and the FIFO advances incorrectly. The WRFULL output should be used to gate the WRFIFO# input to avoid this situation. A similar problem can occur if Add-On logic attempts to read an empty FIFO in 16-bit mode. RDEMPTY should be used to gate the RDFIFO# input to avoid problems with the FIFO getting out of step. In 32-bit mode (MODE = low), these situations do not occur.

If FIFO accesses are done without the direct access signals with MODE configured for 16-bits (using ADR, SELECT#, etc.), external hardware must toggle ADR1 between consecutive 16-bit bus cycles. The FIFO advance condition must be set to correspond to the order the application accesses the upper and lower words in the FIFO register.

CONFIGURATION

The FIFO configuration takes place during initialization and during operation. During initialization, the FIFO hardware interface method and bus master register access rights are defined. During operation, FIFO advance conditions, endian conversion, and bus mastering capabilities are defined. The following section describes the bits and registers which are involved with controlling and monitoring FIFO operation.

FIFO Setup During Initialization

Location 45h in an external non-volatile memory may be used to configure the S5933 FIFO during initialization. If no external non-volatile memory is used, the S5933 defaults to PCI initiated bus master transfers with asynchronous operation for FIFO accesses.

The value of bit 7 in location 45h determines if the address and transfer count registers used in bus mastering are accessible from the PCI bus or from the Add-On bus. Once the configuration information is downloaded from non-volatile memory after reset, the bus mastering initialization method can not be changed. Access to the bus master address and transfer count registers cannot be alternated between the PCI bus and the Add-On interface during operation.

Bits 6 and 5 in location 45h determine if FIFO register accesses using the RDFIFO#, WRFIFO#, RD# and WR# inputs operate asynchronous or synchronous to BPCLK. For asynchronous operation, RDFIFO#, WRFIFO#, RD# and WR# operate as clocks for data. For synchronous operation, RDFIFO#, WRFIFO#, RD# and WR# operate as enables, using BPCLK to clock data. Synchronous operation allows higher data transfer rates.

Location 45h Configuration Bits

Bit 7	Bus Master Register Access
0	Address and transfer count registers only accessible from the Add-On interface
1	Address and transfer count registers only accessible from the PCI interface (default)
Bit 6	RDFIFO#, RD# Operation
0	Synchronous Mode - RDFIFO# and RD# functions as enables
1	Asynchronous Mode - RDFIFO# and RD# functions as clocks (default)
Bit 5	WRFIFO#, WR# Operation
0	Synchronous Mode - WRFIFO# and WR# functions as enables
1	Asynchronous Mode - WRFIFO# and WR# functions as clocks (default)
Bit 0	Target Latency Timer Enable
0	Disable PCI Latency Timer Time Out - Will not disconnect with retry if cannot issue TRDY in specified time
1	Enable PCI Latency Timer Time Out - Will be PCI 2.1 compliant

FIFO Status and Control Bits

The FIFO status can be monitored and the FIFO operation controlled from the PCI Operation Registers and/or the Add-On Operation Registers. The FIFO register resides at offset 20h in the PCI and Add-On Operation Registers.

The Bus Master Control/Status (MCSR) PCI Operation register allows a PCI host to monitor FIFO activity and control FIFO operation. Reset controls allow the PCI to Add-On FIFO and Add-On to PCI FIFO flags to be reset (individually). Status bits indicate if the PCI to Add-On FIFO is empty, has four or more open spaces, or is full. Status bits also indicate if the Add-On to PCI FIFO is empty, has four or more full locations or is full. Finally, FIFO PCI bus mastering is monitored/controlled though this register.

The Add-On General Control/Status (AGCSTS) Add-On Operation Register allows an Add-On CPU to monitor FIFO activity and control FIFO operation. Reset controls allow the PCI to Add-On FIFO and Add-On to PCI FIFO flags to be reset (individually). Status bits indicate if the PCI to Add-On FIFO is empty, has four or more open spaces, or is full. Status bits also indicate if the Add-On to PCI is empty, has four or more full spaces or is full. FIFO bus mastering status may be monitored through this register, but all bus master configuration is through the MCSR PCI Operation Register.

PCI Initiated FIFO Bus Mastering Setup

For PCI initiated bus mastering, the PCI host sets up the S5933 to perform bus master transfers. The following tasks must be completed to setup FIFO bus mastering:

1) Define interrupt capabilities. The PCI to Add-On and/or Add-On to PCI FIFO can generate a PCI interrupt to the host when the transfer count reaches zero.

INTCSR	Bit 15	Enable Interrupt on read transfer count equal zero
INTCSR	Bit 14	Enable Interrupt on write transfer count equal zero

2) Reset FIFO flags. This may not be necessary, but if the state of the FIFO flags is not known, they should be initialized.

MCSR	Bit 26	Reset Add-On to PCI FIFO flags
MCSR	Bit 25	Reset PCI to Add-On FIFO flags

3) Define FIFO management scheme. These bits define what FIFO condition must exist for the PCI bus request (REQ#) to be asserted by the S5933.

MCSR	Bit 13	PCI to Add-On FIFO management scheme
MCSR	Bit 9	Add-On to PCI FIFO management scheme

4) Define PCI to Add-On and Add-On to PCI FIFO priority. These bits determine which FIFO has priority if both meet the defined condition to request the PCI bus. If these bits are the same, priority alternates, with read accesses occurring first.

MCSR	Bit 12	Read vs. write priority
MCSR	Bit 8	Write vs. read priority

5) Define transfer source/destination address. These registers are written with the first address that is to be accessed by the S5933. These address registers are updated after each access to indicate the next address to be accessed. Transfers must start on DWORD boundaries.

MWAR	All	Bus master write address
MRAR	All	Bus master read address

6) Define transfer byte counts. These registers are written with the number of bytes to be transferred. The transfer count does not have to be a multiple of four bytes. These registers are updated after each transfer to reflect the number of bytes remaining to be transferred.

MWTC	All	Write transfer byte count
MRTC	All	Read transfer byte count

7) Enable Bus Mastering. Once steps 1-6 are completed, the FIFO may operate as a PCI bus master. Read and write bus master operation may be independently enabled or disabled.

MCSR	Bit 14	Enable PCI to Add-On FIFO bus mastering
MCSR	Bit 10	Enable Add-On to PCI FIFO bus mastering

The order of the tasks listed above is not particularly important. It is recommended that bus mastering be enabled as the last step. Some applications may choose to leave bus mastering enabled and start transfers by writing a non-zero value to the transfer count registers. This also works, provided the entire transfer count is written in a single access. As a number of the configuration bits and the two enable bits are all in the MCSR register, it may be most efficient for the FIFO configuration bits to be set with the same register access that enables bus mastering.

If interrupts are enabled, a host interrupt service routine is also required. The service routine determines the source of the interrupt and resets the interrupt. As mailbox registers may also be configured to generate interrupts, the exact source of the interrupt is indicated in the PCI Interrupt Control/Status Register (INTCSR). Typically, the interrupt service routine is used to setup the next transfer by writing new addresses and transfer counts, but some applications may also require other actions. If read transfer or write transfer complete interrupts are enabled, master and target abort interrupts are automatically enabled. These indicate a transfer error has occurred. Writing a one to these bits clears the corresponding interrupt.

INTCSR	Bit 21	Target abort caused interrupt
INTCSR	Bit 20	Master abort caused interrupt
INTCSR	Bit 19	Read transfer complete caused interrupt
INTCSR	Bit 18	Write transfer complete caused interrupt

Add-On Initiated FIFO Bus Mastering Setup

For Add-On initiated bus mastering, the Add-On sets up the S5933 to perform bus master transfers. The following tasks must be completed to setup FIFO bus mastering:

1) Define transfer count abilities. For Add-On initiated bus mastering, transfer counts may be either enabled or disabled. Transfer counts for read and write operations cannot be individually enabled.

AGCSTS	Bit 28	Enable transfer count for read and write bus master transfers

2) Define interrupt capabilities. The PCI to Add-On and/or Add-On to PCI FIFO can generate an interrupt to the Add-On when the transfer count reaches zero (if transfer counts are enabled).

AINT	Bit 15	Enable interrupt on read transfer count equal zero
AINT	Bit 14	Enable interrupt on write transfer count equal zero

3) Reset FIFO flags. This may not be necessary, but if the state of the FIFO flags is not known, they should be initialized.

AGCSTS	Bit 25	Reset Add-On to PCI FIFO flags
AGCSTS	Bit 26	Reset PCI to Add-On FIFO flags

4) Define FIFO management scheme. These bits define what FIFO condition must exist for the PCI bus request (REQ#) to be asserted by the S5933. This must be programmed through the PCI interface.

MCSR	Bit 13	PCI to Add-On FIFO management scheme
MCSR	Bit 9	Add-On to PCI FIFO management scheme

5) Define PCI to Add-On and Add-On to PCI FIFO priority. These bits determine which FIFO has priority if both meet the defined condition to request the PCI bus. If these bits are the same, priority alternates, with read accesses occurring first. This must be programmed through the PCI interface.

MCSR	Bit 12	Read vs. write priority
MCSR	Bit 8	Write vs. read priority

6) Define transfer source/destination address. These registers are written with the first address that is to be accessed by the S5933. These address registers are updated after each access to indicate the next address to be accessed. Transfers must start on DWORD boundaries.

MWAR	All	Bus master write address
MRAR	All	Bus master read address

3

7) Define transfer byte counts. These registers are written with the number of bytes to be transferred. The transfer count does not have to be a multiple of four bytes. These registers are updated after each transfer to reflect the number of bytes remaining to be transferred. If transfer counts are disabled, these registers do not need to be programmed.

MWTC	All	Write transfer byte count
MRTC	All	Read transfer byte count

8) Enable Bus Mastering. Once steps 1-7 are completed, the FIFO may operate as a PCI bus master. Read and write bus master operation may be independently enabled or disabled. The AMREN and AMWEN inputs control bus master enabling for Add-On initiated bus mastering. The MCSR bus master enable bits are ignored for Add-On initiated bus mastering.

It is recommended that bus mastering be enabled as the last step. Some applications may choose to leave bus mastering enabled (AMREN and AMWEN asserted) and start transfers by writing a non-zero value to the transfer count registers (if they are enabled).

If interrupts are enabled, an Add-On CPU interrupt service routine is also required. The service routine determines the source of the interrupt and resets the interrupt. As mailbox registers may also be configured to generate interrupts, the exact source of the interrup is indicated in the Add-On Interrupt Control Register (AINT). Typically, the interrupt service routine is used to setup the next transfer by writing new addresses and transfer counts (if enabled), but some applications may also require other actions. If read transfer or write transfer complete interrupts are enabled, the master/target abort interrupt is automatically enabled. These indicate a transfer error has occurred. Writing a one to these bits clears the corresponding interrupt.

AINT	Bit 21	Master/target abort caused interrupt
AINT	Bit 19	Read transfer complete caused interrupt
AINT	Bit 18	Write transfer complete caused interrupt

PASS-THRU OVERVIEW

The S5933 provides a simple registered access port to the PCI bus. Using a handshaking protocol with Add-On card logic, the PCI bus directly accesses resources on the Add-On. The Pass-Thru data transfer method is very useful for direct Add-On memory access, or accessing registers within peripherals on an Add-On board. Pass-Thru operation requires an external nv memory boot device to define and configure the S5933 Pass-Thru regions.

The S5933 provides four user-configurable Pass-Thru regions. Each region corresponds to a PCI Configuration Base Address Register (BADR1-4). A region represents a block of address space (the block size is user-defined). Each block can be mapped into memory or I/O space. Memory mapped regions can request to be located below 1 MByte (Real Mode address space for a PC). Each region also has a configurable bus width for the Add-On bus interface. An 8-, 16-, or 32-bit Add-On interface may be selected, for use with a variety of Add-On memory or peripheral devices.

Pass-Thru features can be used only when the S5933 is a PCI target. As a target, the S5933 Pass-Thru mode supports single data transfers as well as burst transfers. When accessed with burst transfers, the S5933 supports data transfers at the full PCI bandwidth. The data transfer rate is only limited by the PCI initiator performing the access and the speed of the Add-On logic.

FUNCTIONAL DESCRIPTION

To provide the PCI bus Add-On with direct access to Add-On resources, the S5933 has an internal Pass-Thru Address Register (APTA), and a Pass-Thru Data Register (APTD). These registers are connected to both the PCI bus interface and the Add-On bus interface. This allows a PCI initiator to perform Pass-Thru writes (data transferred from the PCI bus to the Add-On bus) or Pass-Thru reads (PCI bus requests data from the Add-On bus). The S5933 Pass-Thru interface supports both single cycle (one data phase) and burst accesses (multiple data phases).

Pass-Thru Transfers

Data transfers between the PCI bus and the Add-On using the Pass-Thru interface are implemented with a handshaking scheme. If the PCI bus writes to an S5933 Pass-Thru region, Add-On logic must read the data from the S5933 and store it on the Add-On. If the PCI bus reads from a Pass-Thru region, Add-On logic must write data to the S5933.

Some applications may require that an address be passed to the Add-On for Pass-Thru accesses. For example, a 4 Kbyte Pass-Thru region on the PCI bus may correspond to a 4 Kbyte block of SRAM on the Add-On card. If a PCI initiator accessed this region, the Add-On would need to know the offset within the memory device to access. The Pass-Thru Address Register (APTA) allows the Add-On to access address information for the current PCI cycle. When the PCI bus performs burst accesses, the APTA register is updated by the S5933 to reflect the address of the current data phase.

For PCI writes to the Add-On, the S5933 transfers the data from the PCI bus into the Pass-Thru Data Register (APTD). The S5933 captures the data from the PCI bus when TRDY# is asserted. The PCI bus then becomes available for other transfers. When the Pass-Thru data register becomes full, the S5933 asserts the Pass-Thru status signals to indicate to the Add-On that data is present. The Add-On logic may read the data register and assert PTRDY# to indicate the current access is complete. Until the current access completes, the S5933 responds to further Pass-Thru accesses with retries.

For PCI reads from the Add-On, the S5933 asserts the Pass-Thru status signals to indicate to the Add-On that data is required. The Add-On logic should write to the Pass-Thru Data Register and assert PTRDY# to complete the access. The S5933 does not assert TRDY# to the PCI bus until PTRDY# is asserted by Add-On logic. If the Add-On cannot provide data quickly enough, the S5933 signals a retry to the PCI bus. This allows the PCI bus to perform other tasks, rather than waiting for a slow target.

Pass-Thru Status/Control Signals

The S5933 Pass-Thru registers are accessed using the standard Add-On register access pins. The Pass-Thru Address Register (APTA) can, optionally, be accessed using a single, direct access input, PTADR#. Pass-Thru cycle status indicators are provided to control Add-On logic based on the type of Pass-Thru access occurring (single cycle, burst, etc.). The following signals are provided for Pass-Thru operation:

Signal	Function
PTATN#	This output indicates a Pass-Thru access is occurring
PTBURST#	This output indicates the Pass-Thru access is a PCI burst access
PTNUM[1:0]	These outputs indicate which Pass-Thru region decoded the PCI address
PTBE[3:0]#	These outputs indicate which data bytes are valid (PCI writes), or requested (PCI reads)
PTWR	This output indicates if the Pass-Thru access is a PCI read or a write
PTADR#	When asserted, this input drives the Pass-Thru Address Register contents onto the Add-On data bus
PTRDY#	When asserted, this input indicates the current Pass-Thru transfer has been completed by the Add-On
BPCLK	Buffered PCI bus clock output (to synchronize Pass-Thru data register accesses)

Pass-Thru Add-On Data Bus Sizing

Many applications require an 8-bit or 16-bit Add-On bus interface. Pass-Thru regions can be configured to support bus widths other than 32-bits. Each Pass-Thru region can be defined, during initialization, as 8, 16-, or 32-bits. All of the regions do not need to be the same. This feature allows a simple interface to 8- and 16-bit Add-On devices.

To support alternate Add-On bus widths, the S5933 performs internal data bus steering. This allows the Add-On interface to assemble and disassemble 32-bit PCI data using multiple Add-On accesses to the Pass-Thru Data Register (APTD). The Add-On byte enable inputs (BE[3:0]#) are used to access the individual bytes or words within APTD.

BUS INTERFACE

The Pass-Thru interface on the S5933 is a PCI target-only function. Pass-Thru operation allows PCI initiators to read or write resources on the Add-On card. A PCI initiator may access the Add-On with single data phase cycles or multiple data phase bursts.

The Add-On interface implements Pass-Thru status and control signals used by logic to complete data transfers initiated by the PCI bus. The Pass-Thru interface is designed to allow Add-On logic to function without knowledge of PCI bus activity. Add-On logic only needs to react to the Pass-Thru status outputs. The S5933 PCI interface independently interacts with the PCI initiator to control data flow between the devices.

The following sections describe the PCI and Add-On bus interfaces. The PCI interface description provides a basic overview of how the S5933 interacts with the PCI bus, and may be useful in system debugging. The Add-On interface description indicates functions required by Add-On logic and details the Pass-Thru handshaking protocol.

PCI Bus Interface

The S5933 decodes all PCI bus cycle addresses. If the address associated with the current cycle is to one of S5933 Pass-Thru regions, DEVSEL# is asserted. If the Pass-Thru logic is currently idle (not busy finishing a previous Pass-Thru operation), the bus cycle type is decoded and the Add-On Pass-Thru status outputs are set to initiate a transfer on the Add-On side. If the Pass-Thru logic is currently busy completing a previous access, the S5933 signals a retry to PCI initiator.

The following sections describe the behavior of the PCI interface for Pass-Thru accesses to the S5933. Single cycle accesses, burst accesses, and target-initiated retries are detailed.

PCI Pass-Thru Single Cycle Accesses

Single cycle transfers are the simplest PCI bus transaction. Single cycle transfers have an address phase and a single data phase. The PCI bus transaction starts when an initiator drives address and command information onto the PCI bus and asserts FRAME#. The initiator always deasserts frame before the last data phase. For single cycle transfers, FRAME# is only asserted during the address phase (indicating the first data phase is also the last).

When the S5933 sees FRAME# asserted, it samples the address and command information to determine if the bus transaction is intended for it. If the address is within one of the defined Pass-Thru regions, the S5933 accepts the transfer (assert DEVSEL#), and stores the PCI address in the Pass-Thru Address Register (APTA).

For Pass-Thru writes, the S5933 responds immediately (asserting TRDY#) and transfers the data from the PCI bus into the Pass-Thru Data Register (APTD). The S5933 then indicates to the Add-On interface that a Pass-Thru write is taking place and waits for Add-On logic to read the APTD register and complete the transfer (assert PTRDY#). Once the S5933 has captured the data from the PCI bus, the transfer is finished from the PCI bus perspective, and the PCI bus becomes available for other transfers.

For Pass-Thru reads, the S5933 indicates to the Add-On interface that a Pass-Thru read is taking place and waits for Add-On logic to write the Pass-Thru Data Register and complete the transfer (assert PTRDY#). The S5933 completes the cycle when data is written into the data register. If the Add-On cannot complete the write quickly enough, the S5933 requests a retry from the initiator. See target-requested disconnect information.

PCI Pass-Thru Burst Accesses

For PCI Pass-Thru burst accesses, the S5933 captures the PCI address and determines if it falls into one of the defined Pass-Thru regions. Accesses that fall into a Pass-Thru region are accepted by asserting DEVSEL#. The S5933 monitors FRAME# and IRDY# on the PCI bus to identify burst accesses. If the PCI initiator is performing a burst access, the Pass-Thru status indicators notify Add-On logic.

For Pass-Thru burst writes, the S5933 responds immediately (asserting TRDY#). The S5933 transfers the first data phase of the burst into the Pass-Thru Data Register (APTD), and stores the PCI address in the Pass-Thru Address Register (APTA). The Add-On interface completes the transfer and asserts PTRDY#. Every time PTRDY# is asserted by the Add-On, the S5933 begins the next data phase. The next data phase is latched into the data register. For burst accesses, APTA is automatically incremented by the S5933 for each data phase.

For Pass-Thru burst reads, the S5933 claims the PCI cycle (asserting DEVSEL#). The request for data is passed on to Add-On logic and the PCI address is stored in the APTA register. The Add-On interface completes the transfer and asserts PTRDY#. The S5933 then drives the requested data on the PCI bus and asserts TRDY# to begin the next data phase. The APTA register is automatically incremented by the S5933 for each data phase.

PCI Retry Conditions

In some applications, Add-On logic may not be able to respond to Pass-Thru accesses quickly. In this situation, the S5933 disconnects from the PCI bus, signaling a retry. This indicates that the initiator should try the access again at a later time. This allows other PCI cycles to be run while the logic on the slow target completes the Pass-Thru access. Ideally, when the initiator retries the access, the target has completed the access and can respond to the initiator.

With many devices, particularly memories, the first access takes longer than subsequent accesses (assuming they are sequential and not random). For this reason, the PCI specification allows 16 clocks to respond to the first data phase of a PCI cycle and 8 clocks for subsequent data phases (in the case of a burst) before a retry must be requested by the S5933.

The S5933 also requests a retry if an initiator attempts to burst past the end of a Pass-Thru region. The S5933 updates the Pass-Thru Address Register (APTA) for each data phase during bursts, and if the updated address is not within the current Pass-Thru region, a retry is requested.

For example, a PCI system may map a 512 byte Pass-Thru memory region to 0DC000h to 0DC1FFh. A PCI initiator attempts a four DWORD burst with a starting address of 0DC1F8h. The first and second data phases complete (filling the DWORDs at 0DC1F8h and 0DC1FCh), but the third data phase causes the S5933 to request a retry. This forces the initiator to present the address 0DC200h on the PCI bus. If this address is part of another S5933 Pass-Thru region, the device accepts the access.

PCI Write Retries

When the S5933 requests a retry for a PCI Pass-Thru write, it indicates that the Add-On is still completing a previous Pass-Thru write access. The Pass-Thru Address and Data Register contents (APTA and APTD) are still required for the previous Pass-Thru operation and cannot be updated by the PCI interface until the access completes (the Add-On asserts PTRDY#).

When the Add-On is busy completing a Pass-Thru write, the S5933 requests an **immediate** retry for all Pass-Thru region accesses, allowing the PCI bus to perform other operations. PCI Operation Registers may be accessed while the Add-On is still completing a Pass-Thru access. Only Pass-Thru region accesses receive retry requests.

PCI Read Retries

When the S5933 requests a retry for a PCI Pass-Thru read, it indicates that the Add-On could not complete the read in the required time. The Pass-Thru data cannot be read by the PCI interface until the Add-On asserts PTRDY#, indicating the access is complete.

If the retry occurs after the Add-On has completed the Pass-Thru operation by writing the appropriate data into the Pass-Thru data register and asserting PTRDY#, the S5933 asserts DEVSEL# and TRDY# to complete the PCI read. If the Add-On still has not completed the Pass-Thru read, the S5933 waits for the required 16 clocks. If the Add-On completes the access during this time, TRDY# is asserted and the access is finished. If the Add-On cannot complete the access within 16 clocks, another retry is requested.

When the Add-On is busy completing a Pass-Thru read, the S5933 requests an **immediate** retry for all Pass-Thru region accesses, except the region currently completing the previous access. This allows the PCI bus to perform other operations. The next access to the Pass-Thru region which initiated the retry must be to the **same address** which caused the retry. Another initiator accessing the same Pass-Thru region causes the S5933 to respond with the original initiator's data (for reads). S5933 PCI Operation Registers may be accessed while the Add-On is still completing a Pass-Thru access. Only other Pass-Thru region accesses receive retry requests.

Add-On Bus Interface

The Pass-Thru address and data registers can be accessed as Add-On operation registers. The interface to the Pass-Thru registers is described in. The Pass-Thru data register is updated on the rising edge of BPCLK. For this reason, all Pass-Thru inputs must be synchronous to BPCLK. In the following sections the Add-On Pass-Thru interface is described for Pass-Thru single cycle accesses, burst accesses, target-requested retries, and when using 8-bit and 16-bit Add-On data buses.

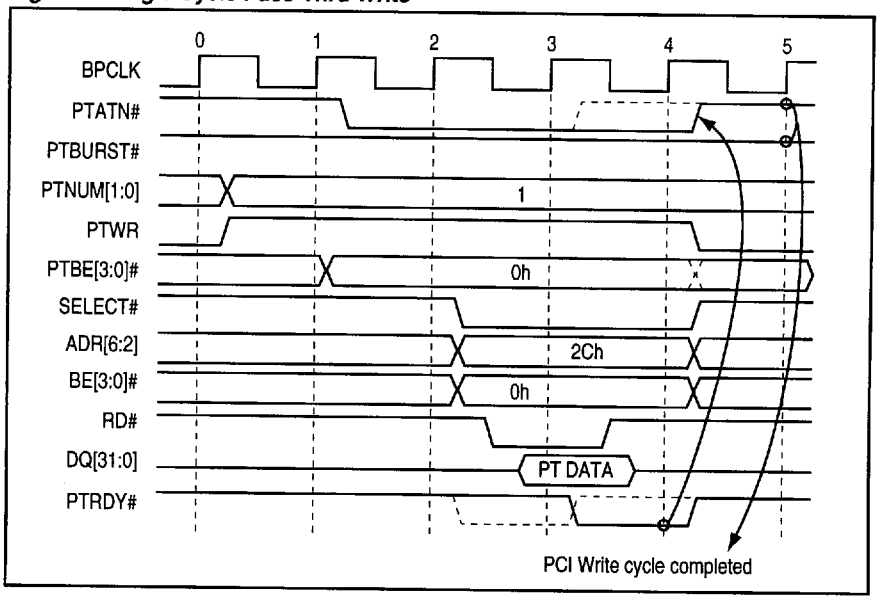
Single Cycle Pass-Thru Writes

A single cycle Pass-Thru write operation occurs when a PCI initiator writes a single value to a Pass-Thru region. PCI single cycle transfers consists of an address phase and one data phase. During the address phase of the PCI transfer, the S5933 stores the PCI address into the Pass-Thru Address Register (APTA). If the S5933 determines that the address is within one of its defined Pass-Thru regions, it captures the PCI data into the Pass-Thru Data Register (APTD).

Figure 1 shows a single cycle Pass-Thru write access (Add-On read). The Add-On must read the data stored in the APTD register and transfer it to its destination. Note: RD# may be asserted for multiple clocks to allow interfacing with slow Add-On devices. Data remains valid until PTRDY# is asserted.

Note:
For all Add-On accesses using PTADR for address data when in 16 bit mode, ADR[1] must be held low to get the low address word.

Figure 1. Single Cycle Pass-Thru Write



Clock 0: The PCI bus cycle address information is stored in the S5933 Pass-Thru Address Register.

Clock 1: The PCI address is recognized as a write to Pass-Thru region 1. The PCI data is stored in the S5933 Pass-Thru Data Register. PTATN# is asserted to indicate a Pass-Thru access is occurring.

Clock 2: Pass-Thru status signals indicate what action is required by Add-On logic. Pass-Thru status outputs are valid when PTATN# is active and are sampled by the Add-On at the rising edge of clock 2.

PTBURST#	Deasserted. The access has a single data phase.
PTNUM[1:0]	01. Indicates the PCI access is to Pass-Thru region 1.
PTWR	Asserted. The Pass-Thru access is a write.
PTBE[3:0]#	0h. Indicates the Pass-Thru access is 32-bits.

SELECT#, address and byte enable inputs are driven to read the Pass-Thru Data Register at offset 2Ch. DQ[31:0] are driven after RD# and SELECT# are asserted.

Clock 3: If PTRDY# is asserted at the rising edge of clock 3, PTATN# is immediately deasserted and the Pass-Thru access is completed at clock 4.

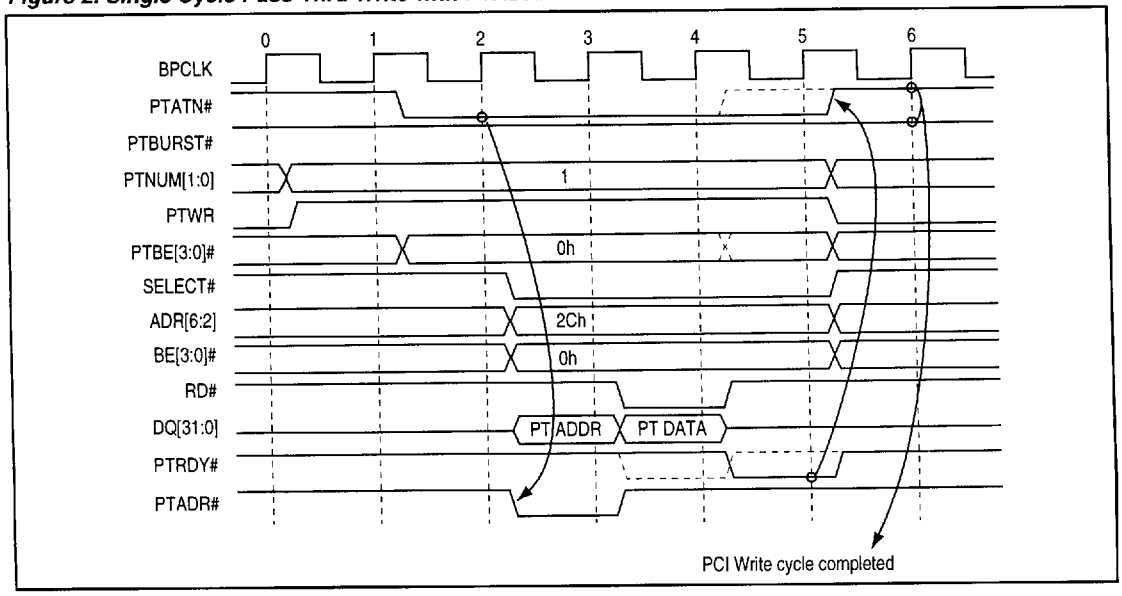
Clock 4: If Add-On logic requires more time to read the Pass-Thru Data Register (slower memory or peripherals), PTRDY# can be delayed, extending the cycle. With PTRDY# asserted at the rising edge of clock 4, PTATN# is deasserted and the Pass-Thru access is completed at clock 5.

Clock 5: PTATN# and PTBURST# deasserted at the rising edge of clock 5 indicates the Pass-Thru access is complete. The S5933 can accept new Pass-Thru accesses from the PCI bus at clock 6.

Figure 2 shows a single cycle Pass-Thru write using the Pass-Thru address information. This provides PCI cycle address information to select a specific address location within an Add-On memory or peripheral. Add-On logic must latch the address for use during the data transfer. Typically, the entire 32-bit address is not required. The Add-On may implement a scheme where only the required number of address bits are latched. It may also be useful to use the Pass-Thru region identifiers, PTNUM[1:0] as address lines. For example, Pass-Thru region 1 might be a 64K block of SRAM for data, while Pass-Thru region 2 might be 64K of SRAM for code storage (downloaded from the host during initialization). Using PTNUM0 as address line A16 allows two unique Add-On memory regions to be defined.

3

Figure 2. Single Cycle Pass-Thru Write with PTADR#



The Add-On PTADR# input directly accesses the Pass-Thru Address Register and drives the contents onto the data bus (no BPCLK rising edge is required). The byte enables, address, and SELECT# inputs are ignored when PTADR# is asserted. RD# and WR# must not be asserted when PTADR# is asserted.

Clock 0: The PCI bus cycle address is stored in the S5933 Pass-Thru Address Register.

Clock 1: The PCI address is recognized as an access to Pass-Thru region 1. PCI data is stored in the S5933 Pass-Thru Data Register. PTATN# is asserted to indicate a Pass-Thru access is occurring.

Clock 2: Pass-Thru status signals indicate what action is required by Add-On logic. Pass-Thru status outputs are valid when PTATN# is active and are sampled by the Add-On at the rising edge of clock 2.

PTBURST#	Deasserted. The access has a single data phase.
PTNUM[1:0]	01. Indicates the PCI access is to Pass-Thru region 1.
PTWR	Asserted. The Pass-Thru access is a write.
PTBE[3:0]#	0h. Indicate the Pass-Thru access is 32-bits.

The PTADR# input is asserted to read the Pass-Thru Address Register. The byte enable, address, and SELECT# inputs are changed during this clock to select the Pass-Thru Data Register during clock cycle 3.

Clock 3: SELECT#, byte enable, and the address inputs remain valid to read the Pass-Thru Data Register at offset 2Ch. RD# is asserted to drive data register contents onto the DQ bus.

Clock 4: If PTRDY# is asserted at the rising edge of clock 4, PTATN# is immediately deasserted and the Pass-Thru access is completed at clock 5.

Clock 5: If Add-On logic requires more time to read the Pass-Thru Data Register (slower memory or peripherals), PTRDY# can be delayed, extending the cycle. PTRDY# asserted at the rising edge of clock 5 causes PTATN# to be immediately deasserted.

Clock 6: PTATN# and PTBURST# deasserted at the rising edge of clock 6 indicates the Pass-Thru access is complete. The S5933 can accept new Pass-Thru accesses from the PCI bus at clock 7.

Single Cycle Pass-Thru Reads

A single cycle Pass-Thru read operation occurs when a PCI initiator reads a single value from a Pass-Thru region. PCI single cycle transfers consists of an address phase and a one data phase. During the address phase of the PCI transfer, the S5933 stores the PCI address into the Pass-Thru Address Register (APTA). If the S5933 determines that the address is within one of its defined Pass-Thru regions, it indicates to the Add-On that a write to the Pass-Thru Data Register (APTD) is required.

Figure 3 shows a single cycle Pass-Thru read access (Add-On write) using PTADR#. The Add-On reads data from a source on the Add-On and writes it to the APTD register.

Clock 0: PCI address information is stored in the S5933 Pass-Thru Address Register. The PCI cycle is recognized as an access to Pass-Thru region 1. PTATN# is asserted by the S5933 to indicate a Pass-Thru access is occurring.

Clock 1: Pass-Thru status signals indicate what action is required by Add-On logic. Pass-Thru status outputs are valid when PTATN# is active and are sampled by the Add-On at the rising edge of clock 1.

PTBURST#	Deasserted. The access has a single data phase.
PTNUM[1:0]	01. Indicates the PCI access was to Pass-Thru region 1.
PTWR	Deasserted. The Pass-Thru access is a read.
PTBE[3:0]#	0h. Indicate the Pass-Thru access is 32-bits.

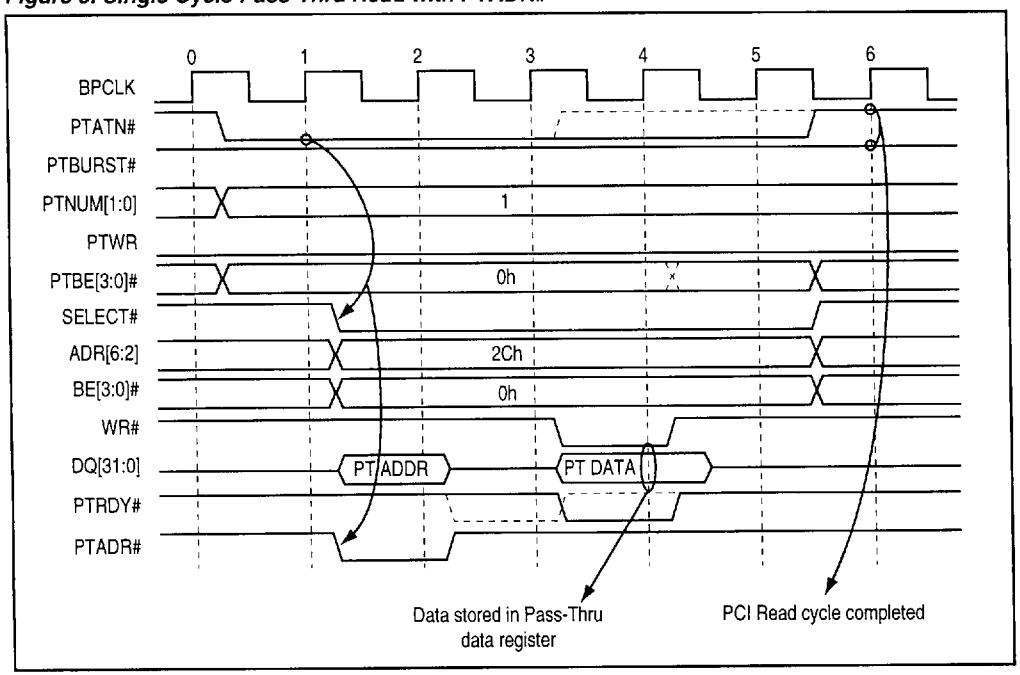
The PTADR# input is asserted to read the Pass-Thru Address Register. The byte enable, address, and SELECT# inputs are changed during this clock to select the Pass-Thru Data Register during clock cycle 3.

Clock 2: This clock is required to avoid contention on the DQ bus. Time must be allowed after PTADR# is deasserted for the DQ outputs to float before Add-On logic attempts to write to the Pass-Thru Data Register.

Clock 3: SELECT#, byte enables, and the address inputs remain valid to write the Pass-Thru Data Register at offset 2Ch. If WR# is asserted at the rising edge of clock 3, data on the DQ bus is latched into APTD.

If PTRDY# is asserted at the rising edge of clock 3, PTATN# is immediately deasserted and the Pass-Thru access is completed at clock 4.

Figure 3. Single Cycle Pass-Thru Read with PTADR#



Clock 4: If Add-On logic requires more time to write the Pass-Thru data register (slower memory or peripherals), PTRDY# can be delayed, extending the cycle. PTRDY# asserted at the rising edge of clock 4 causes PTATN# to be immediately deasserted and the Pass-Thru access is completed at clock 5.

Clock 5: PTATN# and PTBURST# deasserted at the rising edge of clock 5 indicates the Pass-Thru access is complete. The S5933 can accept new Pass-Thru accesses from the PCI bus at clock 6.

Pass-Thru Burst Writes

A Pass-Thru burst write operation occurs when a PCI initiator writes multiple values to a Pass-Thru region. A PCI burst cycle consists of an address phase and multiple data phases. During the address phase of the PCI transfer, the S5933 stores the PCI address into the Pass-Thru Address Register (APTA). If the S5933 determines that the address is within one of its defined Pass-Thru regions, it captures the PCI data into the Pass-Thru Data Register (APTD). After the Add-On completes each read from the Pass-Thru data register (asserts PTRDY#), the next data phase is initiated.

Figure 4 shows a 6 data phase Pass-Thru burst write (Add-On read). In this case, the Add-On asserts PTADR# and then reads multiple data phases from the S5933. This works well for Add-On logic which supports burst cycles. If the Add-On logic does not support burst accesses, PTADR# may be pulsed before each data phase. The S5933 automatically increments the address in the APTA register during PCI burst cycles. In this example PTRDY# is always asserted, indicating Add-On logic is capable of accepting data at a rate of one DWORD per clock cycle.

Clock 0: PCI address information is stored in the S5933 Pass-Thru Address Register.

Clock 1: The PCI address is recognized as an access to Pass-Thru region 1. PCI data for the first data phase is stored in the S5933 Pass-Thru Data Register. PTATN# is asserted by the S5933 to indicate a Pass-Thru access is occurring.

Clock 2: Pass-Thru status signals indicate what action is required by Add-On logic. Pass-Thru status outputs are valid when PTATN# is active and are sampled by the Add-On at the rising edge of clock 2.

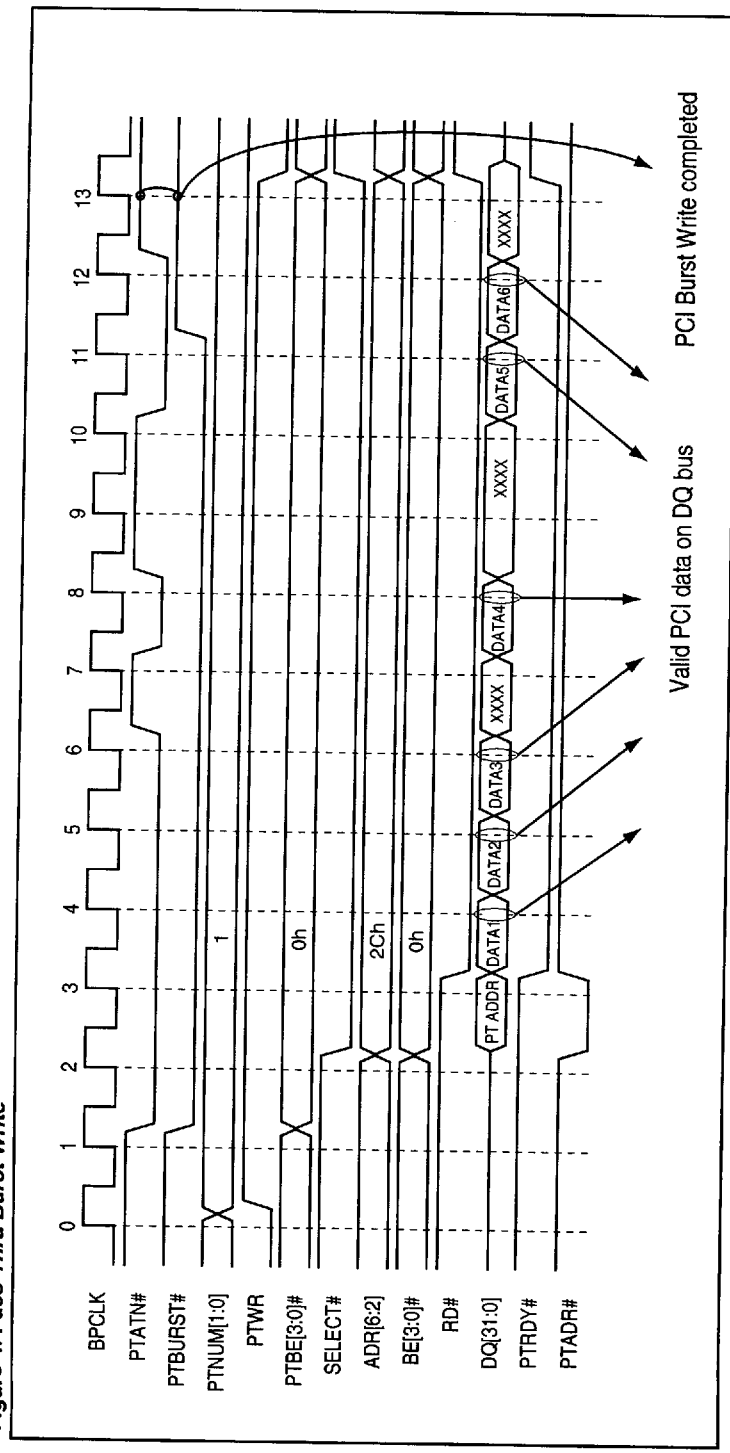


Figure 4. Pass-Thru Burst Write

PTBURST#	Asserted. The access has a multiple data phases.
PTNUM[1:0]	01. Indicates the PCI access was to Pass-Thru region 1.
PTWR	Asserted. The Pass-Thru access is a write.
PTBE[3:0]#	0h. Indicate the Pass-Thru access is 32-bits.

The PTADR# input is asserted to read the Pass-Thru Address Register. The byte enable, address, and SELECT# inputs are changed during this clock to select the Pass-Thru Data Register during clock cycle 3.

Clock 3: SELECT#, byte enables, and the address inputs remain driven to read the Pass-Thru Data Register at offset 2Ch. RD# is asserted to drive data register contents onto the DQ bus.

Clock 4: Add-On logic uses the rising edge of clock 4 to store DATA 1 from the S5933. PTRDY# asserted at the rising edge of clock 4 completes the current data phase. DATA 2 is driven on the Add-On bus.

Clock 5: Add-On logic uses the rising edge of clock 5 to store DATA 2 from the S5933. PTRDY# asserted at the rising edge of clock 5 completes the current data phase. DATA 3 is driven on the Add-On bus.

Clock 6: Add-On logic uses the rising edge of clock 6 to store DATA 3 from the S5933. PTRDY# asserted at the rising edge of clock 6 completes the current data phase. On the PCI bus, IRDY# has been deasserted, causing PTATN# to be deasserted. This is how a PCI initiator adds wait states, if it cannot provide data quickly enough. Data on the Add-On bus is not valid.

Clock 7: Because PTATN# remains deasserted, Add-On logic cannot store data at the rising edge of clock 7. PTATN# is reasserted, indicating the PCI initiator is no longer adding wait states. DATA 4 is driven on the Add-On bus.

Clock 8: Add-On logic uses the rising edge of clock 8 to store DATA 4 from the S5933. PTRDY# asserted at the rising edge of clock 8 completes the current data phase. On the PCI bus, IRDY# has been deasserted again, causing PTATN# to be deasserted. Data on the Add-On bus is not valid.

Clock 9: The PCI initiator is still adding wait states. Add-On logic cannot store data while PTATN# is deasserted.

Clock 10: Because PTATN# remains deasserted, Add-On logic cannot read data at the rising edge of clock 10. PTATN# is reasserted, indicating the PCI initiator is no longer adding wait states. DATA 5 is driven on the Add-On bus.

Clock 11: Add-On logic uses the rising edge of clock 11 to store DATA 5 from the S5933. PTRDY# asserted at the rising edge of clock 11 completes the current data phase. DATA 6 is driven on the Add-On bus.

Clock 12: Add-On logic uses the rising edge of clock 12 to store DATA 6 from the S5933. PTRDY# asserted at the rising edge of clock 12 completes the final data phase

Clock 13: PTATN# and PTBURST# deasserted at the rising edge of clock 13 indicates the Pass-Thru access is complete. The S5933 can accept new Pass-Thru accesses from the PCI bus at clock 15.

Figure 5 also shows a 5 data phase Pass-Thru burst write, but the Add-On logic uses PTRDY# to control the rate at which data is transferred. In many applications, Add-On logic is not fast enough to accept data at every BPCLK rising edge (every 30 ns in a 33 MHz PCI system). In this example, the Add-On interface accepts data every other clock. In the example, RD# is asserted during the entire Add-On burst, but it can be deasserted when PTRDY# is deasserted, the S5933 functions the same under both conditions.

Clock 0: PCI address information is stored in the S5933 Pass-Thru Address Register.

Clock 1: The PCI address is recognized as an access to Pass-Thru region 1. PCI data for the first data phase is stored in the S5933 Pass-Thru Data Register. PTATN# is asserted by the S5933 to indicate a Pass-Thru access is occurring.

Clock 2: Pass-Thru status signals indicate what action is required by Add-On logic. Pass-Thru status outputs are valid when PTATN# is active and are sampled by the Add-On at the rising edge of clock 2.

PTBURST#	Asserted. The access has multiple data phases.
PTNUM[1:0]	01. Indicates the PCI access is to Pass-Thru region 1.
PTWR	Asserted. The Pass-Thru access is a write.
PTBE[3:0]#	0h. Indicate the Pass-Thru access is 32-bits.

The PTADR# input is asserted to read the

3

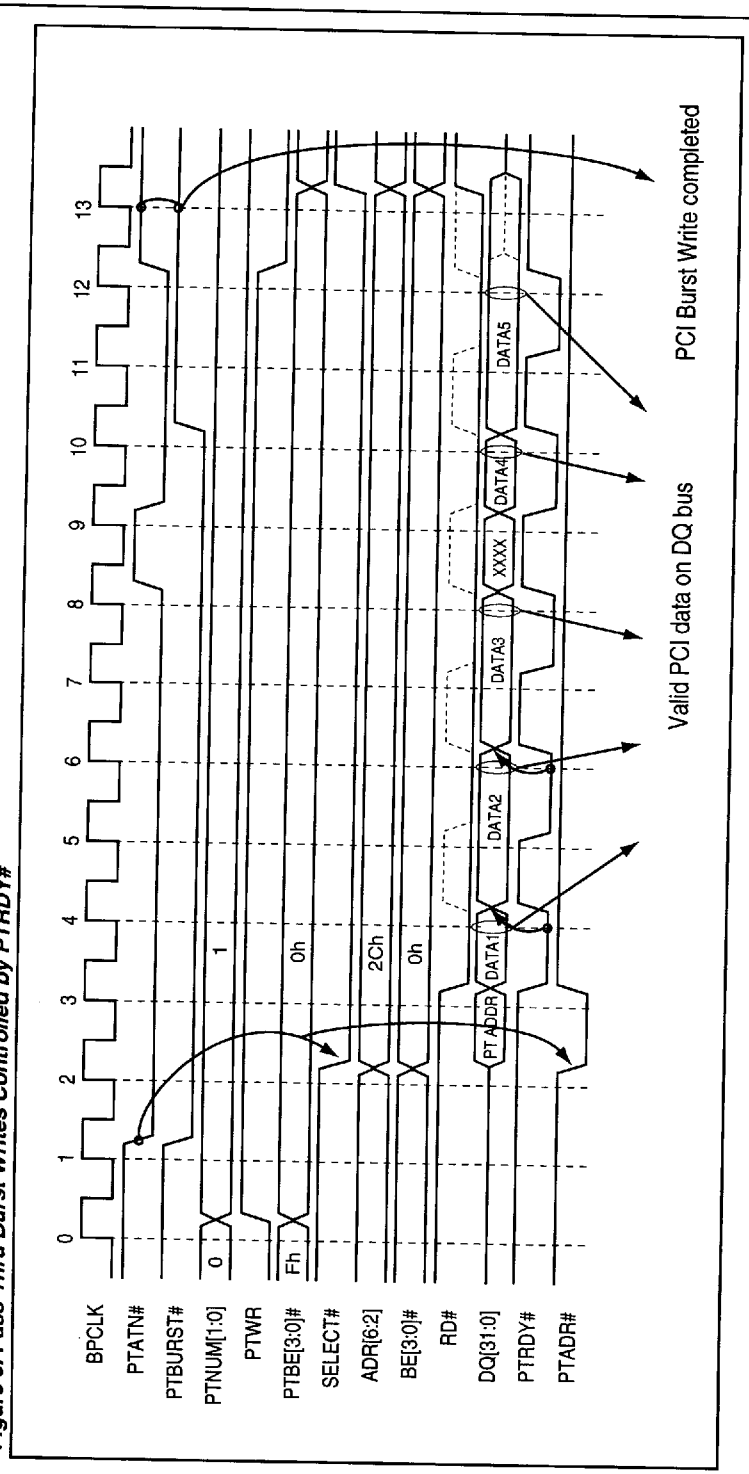


Figure 5. Pass-Thru Burst Writes Controlled by PTRDY#

Pass-Thru Address Register. The byte enable, address, and SELECT# inputs are changed during this clock to select the Pass-Thru Data Register during clock cycle 3.

Clock 3: SELECT#, byte enable, and the address inputs remain driven to read the Pass-Thru Data Register at offset 2Ch. RD# is asserted to drive data register contents onto the Add-On data bus.

Clock 4: Add-On logic uses the rising edge of clock 4 to store DATA 1 from the S5933. PTRDY# asserted at the rising edge of clock 4 completes the current data phase. DATA 2 is driven on the Add-On bus.

Clock 5: Add-On logic is not fast enough to store DATA 2 by the rising edge of clock 5. PTRDY# deasserted at the rising edge of clock 5 extends the current data phase and DATA 2 remains driven on the Add-On bus.

Clock 6: Add-On logic uses the rising edge of clock 6 to store DATA 2 from the S5933. PTRDY# asserted at the rising edge of clock 6 completes the current data phase. DATA 3 is driven on the Add-On bus.

Clock 7: Add-On logic is not fast enough to store DATA 3 by the rising edge of clock 7. PTRDY# deasserted at the rising edge of clock 7 extends the current data phase is and DATA 3 remains driven on the Add-On bus.

Clock 8: Add-On logic uses the rising edge of clock 8 to store DATA 3 from the S5933. PTRDY# asserted at the rising edge of clock 8 completes the current data phase. On the PCI bus, IRDY# has been deasserted, causing PTATN# to be deasserted. Data on the Add-On bus is not valid.

Clock 9: Because PTATN# remains deasserted, Add-On logic cannot store data at the rising edge of clock 9. PTATN# is reasserted, indicating the PCI initiator is no longer adding wait states. DATA 4 is driven on the Add-On bus

Clock 10: Add-On logic uses the rising edge of clock 10 to store DATA 4 from the S5933. PTRDY# asserted at the rising edge of clock 10 completes the current data phase. DATA 5 is driven on the Add-On bus. PTBURST# is deasserted, indicating that on the PCI bus, the burst is complete except for the last data phase. Since the data is double buffered, there may be one or two pieces of data available to the Add-On when PTBURST# becomes inactive. This example shows the single data available case. If another piece of data was available, then PTATN# would remain active instead of going inactive at clock 12.

Clock 11: Add-On logic is not fast enough to store DATA 5 by the rising edge of clock 11. PTRDY# deasserted at the rising edge of clock 11 extends the data phase and DATA 5 remains driven on the Add-On bus.

Clock 12: Add-On logic uses the rising edge of clock 12 to store DATA 5 from the S5933. PTRDY# asserted at the rising edge of clock 12 completes the final data phase.

Clock 13: PTATN# deasserted at the rising edge of clock 13 indicates the Pass-Thru access is complete. The S5933 can accept new Pass-Thru accesses from the PCI bus at clock 14.

Pass-Thru Burst Reads

A Pass-Thru burst read operation occurs when a PCI initiator reads multiple DWORDs from a Pass-Thru region. A burst transfer consists of a single address and a multiple data phases. During the address phase of the PCI transfer, the S5933 stores the PCI address into the Pass-Thru Address Register (APTA). If the S5933 determines that the address is within one of its defined Pass-Thru regions, it indicates to the Add-On that a write to the Pass-Thru Data Register (APTD) is required. Figure 6 shows a 6 data phase Pass-Thru burst read access (Add-On write) using PTADR#.

Clock 0: PCI address information is stored in the S5933 Pass-Thru Address Register. The PCI address is recognized as an access to Pass-Thru region 1. PTATN# is asserted by the S5933 to indicate a Pass-Thru access is occurring.

Clock 1: Pass-Thru status signals indicate what action is required by Add-On logic. Pass-Thru status outputs are valid when PTATN# is active and are sampled by the Add-On at the rising edge of clock 2.

PTBURST#	Deasserted, the S5933 does not yet recognize a PCI burst.
PTNUM[1:0]	01. Indicates the PCI access is to Pass-Thru region 1.
PTWR	Deasserted. The Pass-Thru access is a read.
PTBE[3:0]#	0h. Indicate the Pass-Thru access is 32-bits.

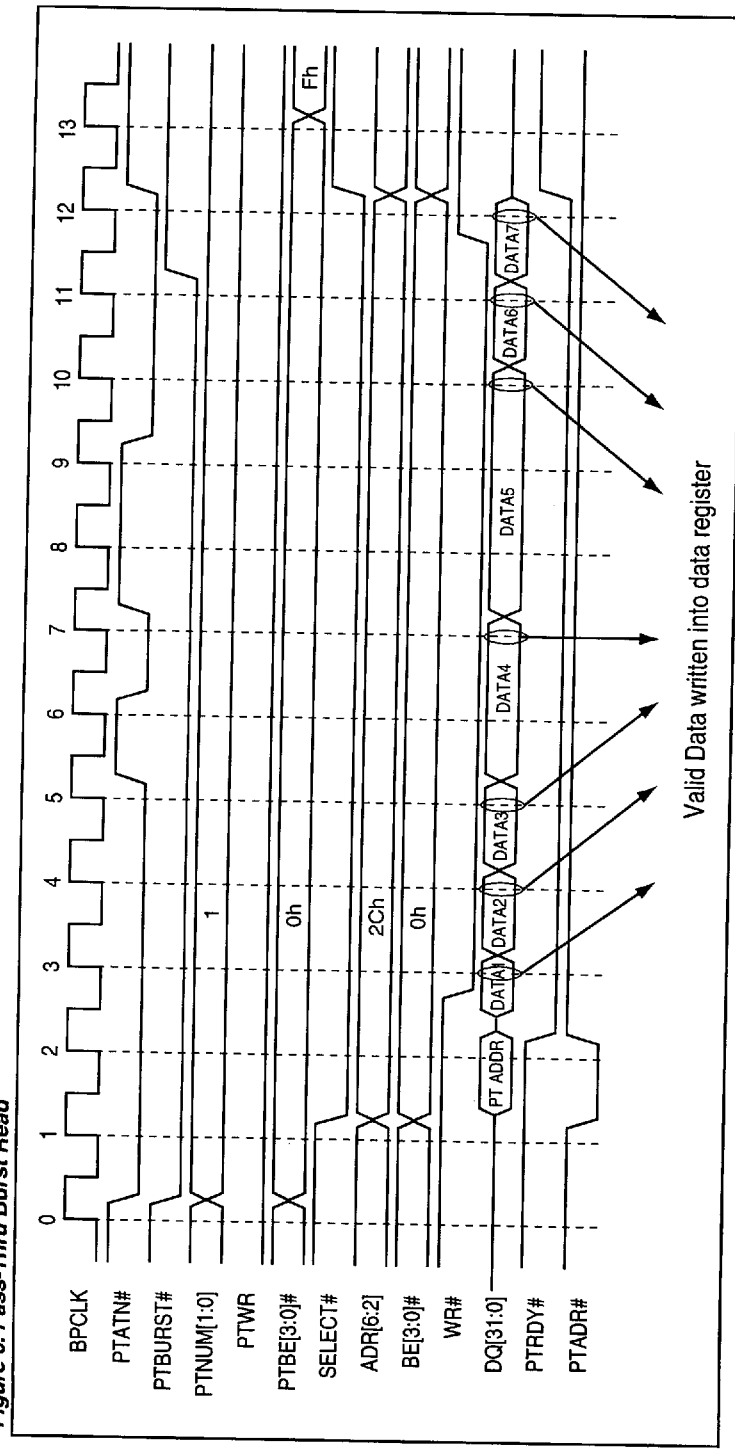


Figure 6. Pass-Thru Burst Read

The PTADR# input is asserted to read the Pass-Thru Address Register. The byte enable, address, and SELECT# inputs are changed during this clock to select the Pass-Thru Data Register during clock cycle 3.

Clock 2: SELECT#, byte enables, and the address inputs remain driven to read the Pass-Thru Data Register at offset 2Ch. PTBURST# is asserted by the S5933, indicating the current Pass-Thru read is a burst.

Clock 3: WR# asserted at the rising edge of clock 3 writes DATA 1 into the S5933. PTRDY# asserted at the rising edge of clock 3 completes the current data phase.

Clock 4: WR# asserted at the rising edge of clock 4 writes DATA 2 into the S5933. PTRDY# asserted at the rising edge of clock 4 completes the current data phase.

Clock 5: WR# asserted at the rising edge of clock 5 writes DATA 3 into the S5933. PTRDY# asserted at the rising edge of clock 5 completes the current data phase. On the PCI bus, IRDY# has been deasserted, causing PTATN# to be deasserted. This is how a PCI initiator adds wait states, if it cannot read data quickly enough.

Clock 6: PTATN# remains deasserted at the rising edge of clock 6. The Add-On cannot write DATA 4 until PTATN# is asserted. PTATN# is reasserted during the cycle, indicating the PCI initiator is no longer adding wait states. Add-On logic continues to drive DATA 4 on the Add-On bus.

Clock 7: WR# asserted at the rising edge of clock 7 writes DATA 4 into the S5933. PTRDY# asserted at the rising edge of clock 7 completes the current data phase. On the PCI bus, IRDY# has been deasserted, causing PTATN# to be deasserted. The PCI initiator is adding wait states.

Clock 8: PTATN# remains deasserted at the rising edge of clock 8. The Add-On cannot write DATA 5 until PTATN# is asserted. Add-On logic continues to drive DATA 5 on the Add-On bus.

Clock 9: PTATN# remains deasserted at the rising edge of clock 9. The Add-On cannot write DATA 5 until PTATN# is asserted. Add-On logic continues to drive DATA 5 on the Add-On bus. PTATN# is reasserted during the cycle, indicating the PCI initiator is done adding wait states.

Clock 10: WR# asserted at the rising edge of clock 10 writes DATA 5 into the S5933. PTRDY# asserted at the rising edge of clock 10 completes the current data phase.

Clock 11: WR# asserted at the rising edge of clock 11 writes DATA 6 into the S5933. PTRDY# asserted at the rising edge of clock 11 completes the final data phase.

Clock 12: PTBURST# is deasserted at the rising edge of clock 12 indicating the Pass-Thru burst is complete. The S5933 can accept new Pass-Thru accesses from the PCI bus at clock 14. Any data written into the Pass-Thru data register is not required and is never passed to the PCI interface (as PTRDY# is not asserted at the rising edge of clock 13).

Figure 7 also shows a 5 data phase Pass-Thru burst read, but the Add-On logic uses PTRDY# to control the rate at which data is transferred. In many applications, Add-On logic is not fast enough to provide data every BPCLK (every 30 ns in a 33 MHz PCI system). In this example, the Add-On interface writes data every other clock cycle. WR# is shown asserted during the entire Add-On burst, but WR# can be deasserted when PTRDY# is deasserted, the S5933 functions the same under both conditions.

Clock 0: PCI address information is stored in the S5933 Pass-Thru Address Register. The PCI address is recognized as an access to Pass-Thru region 1. PTATN# is asserted by the S5933 to indicate a Pass-Thru access is occurring.

Clock 1: Pass-Thru status signals indicate what action is required by Add-On logic. Pass-Thru status outputs are valid when PTATN# is active and are sampled by the Add-On at the rising edge of clock 2.

PTBURST#	Deasserted, the S5933 does not yet recognize a PCI burst.
PTNUM[1:0]	01. Indicates the PCI access is to Pass-Thru region 1.
PTWR	Deasserted. The Pass-Thru access is a read.
PTBE[3:0]#	0h. Indicate the Pass-Thru access is 32-bits.

The PTADR# input is asserted to read the Pass-Thru Address Register. The byte enable, address, and SELECT# inputs are changed during this clock to select the Pass-Thru Data Register during clock cycle 3.

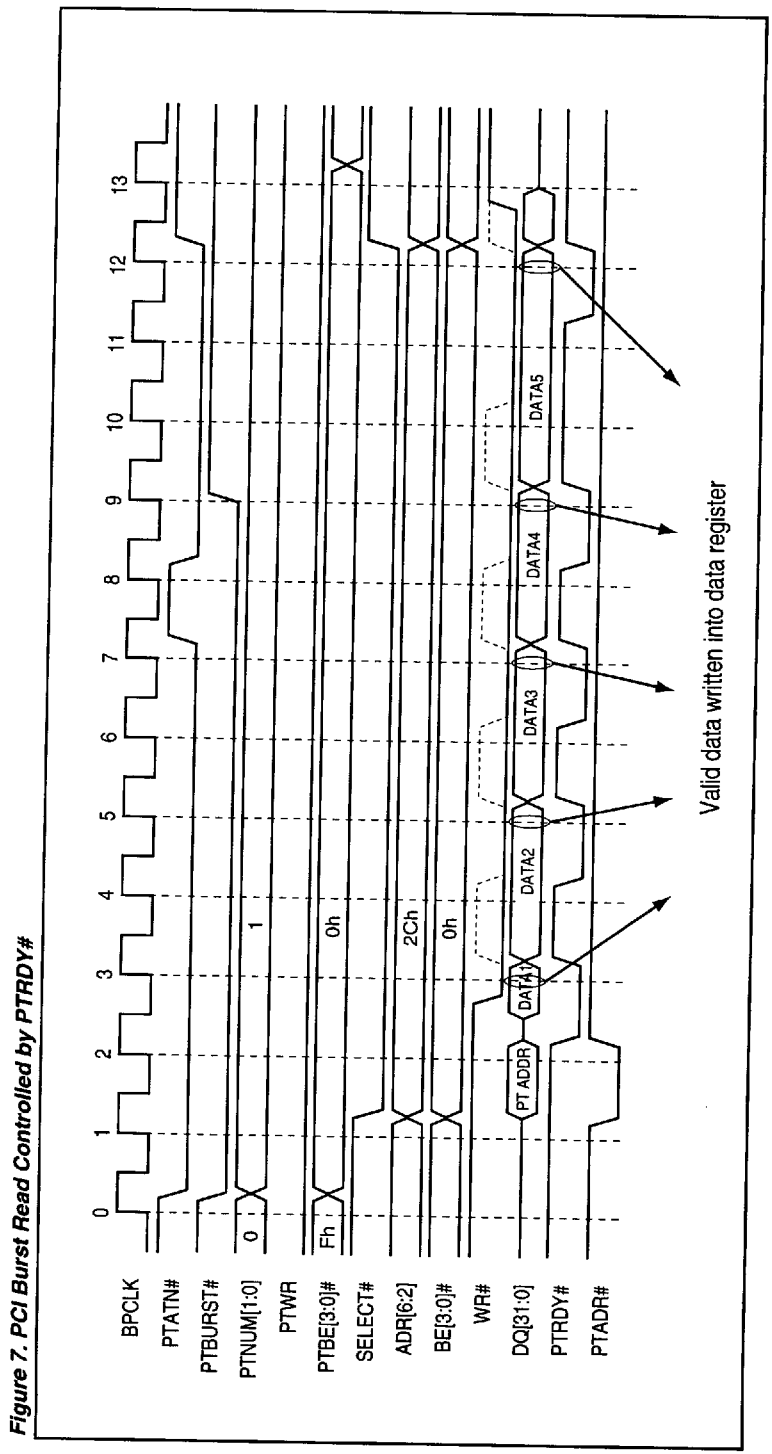


Figure 7. PCI Burst Read Controlled by PTRDY#

Clock 2: SELECT#, byte enable, and the address inputs remain driven to read the Pass-Thru Data Register at offset 2Ch. PTBURST# is asserted by the S5933, indicating the current Pass-Thru read is a burst.

Clock 3: WR# asserted at the rising edge of clock 3 writes DATA 1 into the S5933. PTRDY# asserted at the rising edge of clock 3 completes the current data phase.

Clock 4: Add-On logic drives DATA 2 on the Add-On bus, but PTRDY# deasserted at the rising edge of clock 4 extends the current data phase.

Clock 5: WR# asserted at the rising edge of clock 5 writes DATA 2 into the S5933. PTRDY# asserted at the rising edge of clock 5 completes the current data phase.

Clock 6: Add-On logic drives DATA 3 on the Add-On bus, but PTRDY# deasserted at the rising edge of clock 6 extends the current data phase.

Clock 7: WR# asserted at the rising edge of clock 7 writes DATA 3 into the S5933. PTRDY# asserted at the rising edge of clock 7 completes the current data phase. On the PCI bus, IRDY# has been deasserted, causing PTATN# to be deasserted. This is how a PCI initiator adds wait states, if it cannot read data quickly enough.

Clock 8: PTATN# remains deasserted at the rising edge of clock 8. The Add-On cannot write DATA 4 until PTATN# is asserted. Add-On logic continues to drive DATA 4 on the Add-On bus. PTATN# is reasserted during the cycle, indicating the PCI initiator is done adding wait states.

Clock 9: WR# asserted at the rising edge of clock 9 writes DATA 4 into the S5933. PTRDY# asserted at the rising edge of clock 9 completes the current data phase.

Clock 10: Add-On logic drives DATA 5 on the Add-On bus, but PTRDY# deasserted at the rising edge of clock 10 extends the current data phase.

Clock 11: PTATN# remains deasserted at the rising edge of clock 11. The Add-On does not have to write DATA 5 until PTATN# is asserted. Add-On logic continues to drive DATA 5 on the Add-On bus. PTATN# is reasserted during the cycle, indicating the PCI initiator is done adding wait states.

Clock 12: PTRDY# asserted at the rising edge of clock 12 completes the final data phase. Any data written into the Pass-Thru data register is not required and is never passed to the PCI interface (as PTRDY# is not asserted at the rising edge of clock 13).

Clock 13: PTATN# and PTBURST# deasserted at the rising edge of clock 13 indicates the Pass-Thru access is complete. The S5933 can accept new Pass-Thru accesses from the PCI bus at clock 14.

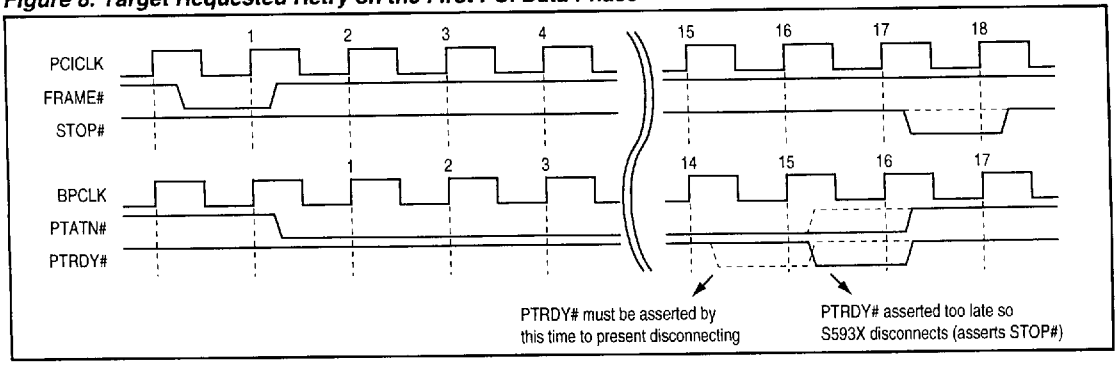
Add-On Pass-Thru Disconnect Operation

Slow PCI targets are prevented from degrading PCI bus performance. The PCI specification allows only 16 clocks for a target to respond before it must request a retry on single data phase accesses. For burst accesses, the first data phase is allowed 16 clocks to complete, all subsequent data phases are allowed 8 clocks each. This requirement allows other PCI initiators to use the bus while the target requesting the retry completes the original access.

Figure 8 shows the conditions that cause the S5933 to request a retry from a PCI initiator on the first data phase of a PCI read operation. FRAME# is asserted during the rising edge of PCI clock 1. From this point,

3

Figure 8. Target Requested Retry on the First PCI Data Phase



the S5933 has 16 clock cycles to respond to the initiator with TRDY# (completing the cycle). FRAME# could remain asserted, indicating a burst read, but the retry request conditions are identical for a single data phase read and the first data phase of a burst read.

BPCLK is identical to PCICLK, lagging by a propagation delay of a few nanoseconds (see Chapter 13). PTATN# is asserted on the Add-On interface as soon as FRAME# is sampled active at a PCICLK rising edge.

After PTATN# is asserted, PTRDY# must be asserted by the 15th BPCLK rising edge to prevent the S5933 from requesting a retry. TRDY# is asserted on the PCI interface one clock cycle after PTRDY# is asserted on the Add-On interface. If Add-On logic does not return PTRDY# by the 15th BPCLK rising edge, the S5933 asserts STOP#, requesting a retry from the PCI initiator.

For Pass-Thru write operations, the S5933 never disconnects on the first or second PCI data phases of a burst. The first data and second phases are always accepted immediately by the S5933. No further action is required by the PCI initiator. The only situation where the S5933 may respond to a Pass-Thru write with a retry request is after the second data phase of a Pass-Thru burst write.

Figure 9 shows the conditions required for the S5933 to request a retry after the second data phase of a burst transfer. This figure applies to both Pass-Thru burst read and write operations.

The previous data phase is completed with the assertion of PTRDY# at the rising edge of BPCLK 0. Add-On logic must assert PTRDY# by the rising edge of BPCLK 8 to prevent the S5933 from asserting STOP#, requesting a retry. Meeting this condition allows the S5933 to assert TRDY# by the rising edge of PCICLK 8, completing the data phase with requiring a retry.

When the S5933 requests a retry, the Pass-Thru status indicators remain valid (allowing the Add-On logic to complete the access). PTBURST# is the exception to this. PTBURST# is deasserted to indicate that there is currently no burst in progress on the PCI bus. The other Pass-Thru status indicators remain valid until PTATN# is deasserted. Figure 10 shows the Add-On bus interface signals after the S5933 requests a retry.

As long as PTATN# remains asserted, Add-On logic should continue to transfer data. For PCI read operations, one Add-On write operation is required after a retry request. After the Add-On write, asserting PTRDY# deasserts PTATN#.

For Pass-Thru write operations, one or two data transfers may remain after the S5933 signals a retry. Two data transfers are possible because the S5933 has a double buffered Pass-Thru data register used for writes. A PCI burst may have filled both registers before the S5933 requested a retry. PTATN# remains asserted until both are emptied. PTRDY# must be asserted after each read from the Pass-Thru data register. If both registers are full, PTATN# is deasserted only after PTRDY# is asserted the second time. The S5933 only accepts further PCI accesses after both registers are emptied.

8-Bit and 16-Bit Pass-Thru Add-On Bus Interface

The S5933 allows a simple interface to devices with 8-bit or 16-bit data buses. Each Pass-Thru region may be defined as 8-, 16-, or 32-bits, depending on the contents of the nv memory boot device loaded into the PCI Base Address Configuration Registers during initialization. The Pass-Thru Add-On interface internally controls byte lane steering to allow access to the 32-bit Pass-Thru Data Register (APTD) from 8-bit or 16-bit Add-On buses.

Figure 9. Target Requested Retry after the First Data Phase of a Burst Operation

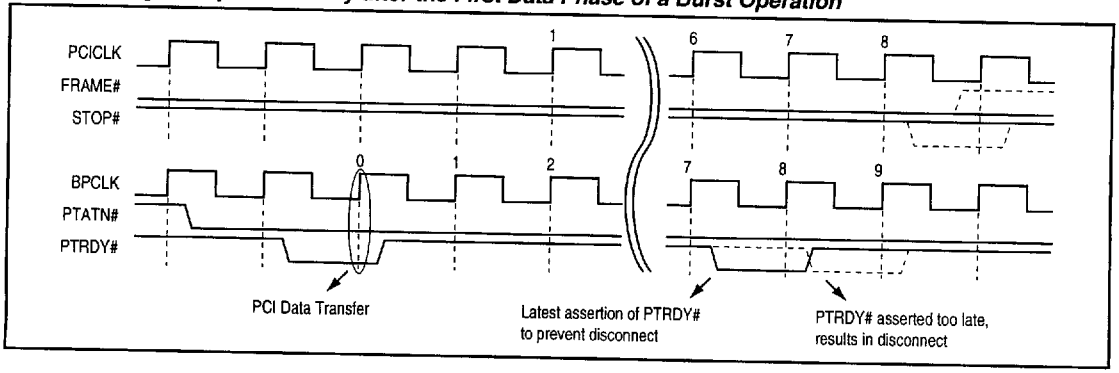
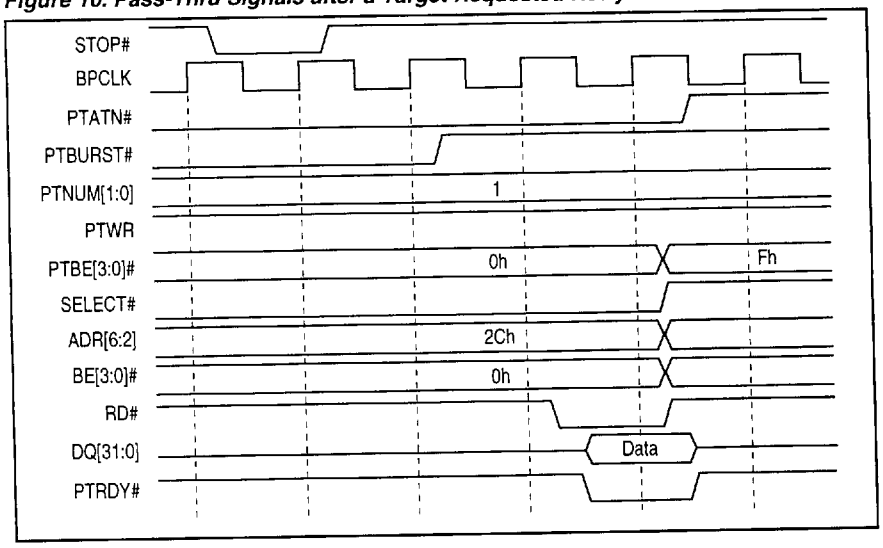


Figure 10. Pass-Thru Signals after a Target Requested Retry



Internal byte lane steering may be used whether the MODE input defines a 16-bit or 32-bit Add-On interface. When a 16-bit Add-On interface is used, the ADR1 input is used in conjunction with the byte enables to steer data into the proper APTD register byte locations.

If MODE defines a 16-bit interface, only 16-bits of address are driven when PTADR# is asserted. If more than 16-bits of address are required, the Pass-Thru Address Register (APTA) must be read with SELECT#, RD#, byte enable and address inputs. Two consecutive reads are required to latch all of the address information (one with ADR1=0, one with ADR1=1).

Regardless of MODE, various data widths may be used. For Pass-Thru writes (Add-On APTD reads), Add-On logic must read the APTD register one byte or one word at a time (depending on the Add-On bus width). The internal data bus is steered to the correct portion of APTD using the BE[3:0]# inputs. Table 1 shows the byte lane steering mechanism used by the S5933. The BYTEn symbols indicate data bytes in the Pass-Thru Data Register.

When a read is performed with a BEn# input asserted, the corresponding PTBEn# output is deasserted. Add-On logic cycles through the byte enables to read the entire APTD register. Once all data is read (PTBE[3:0]# are deasserted), PTRDY# is asserted by the Add-On, completing the access.

For Pass-Thru reads (Add-On APTD writes), the bytes requested by the PCI initiator are indicated by the PTBE[3:0]# outputs. Add-On logic uses the PTBE[3:0]# signals to determine which bytes must be written (and which bytes have already been written). For example, a

Table 1. Byte Lane Steering for Pass-Thru Data Register Read (PCI Write)

Byte Enables				APTD Register Read Byte Lane Steering			
3	2	1	0	DQ[31:24]	DQ[23:16]	DQ[15:8]	DQ[7:0]
x	x	x	0	BYTE3	BYTE2	BYTE1	BYTE0
x	x	0	1	BYTE3	BYTE2	BYTE1	BYTE1
x	0	1	1	BYTE3	BYTE2	BYTE3	BYTE2
0	1	1	1	BYTE3	BYTE3	BYTE3	BYTE3

PCI initiator performs a byte Pass-Thru read from an 8-bit Pass-Thru region with PCI BE2# asserted. On the Add-On interface, PTBE2# is asserted, indicating that the PCI initiator requires data in this byte. Once the Add-On writes APTD, byte 2, PTBE2# is deasserted, and the Add-On may assert PTRDY#, completing the cycle.

Table 2 shows how the external Add-On data bus is steered to the Pass-Thru Data Register bytes. This mechanism is determined by the Pass-Thru region bus width defined during initialization (see Section 12.3). The BYTEn symbols indicate data bytes in the Pass-Thru Data Register. For example, an 8-bit Add-On write with BE1# asserted results in the data on DQ[7:0] being steered into BYTE1 of the APTD register.

Table 2. Byte Lane Steering for Pass-Thru Data Register Write (PCI Read)

Defined	APTD Register Write Byte Lane Steering			
PT-Bus Width	BYTE3	BYTE2	BYTE1	BYTE0
32-Bit Data Bus	DQ[31:24]	DQ[23:16]	DQ[15:8]	DQ[7:0]
16-Bit Data Bus	DQ[15:8]	DQ[7:0]	DQ[15:8]	DQ[7:0]
8-Bit Data Bus	DQ[7:0]	DQ[7:0]	DQ[7:0]	DQ[7:0]

To write data into the APTD Register, the PTBEn# output and the BEn# input must both be asserted. The following describes how APTD Register writes are controlled:

Write BYTE3 if PTBE3# AND BE3# are asserted

Write BYTE2 if PTBE2# AND BE2# are asserted

Write BYTE1 if PTBE1# AND BE1# are asserted

Write BYTE0 if PTBE0# AND BE0# are asserted

After each byte is written into the Pass-Thru data register, its corresponding PTBE[3:0]# output is deasserted. This allows Add-On logic to monitor which bytes have been written, and which bytes remain to be written. When all bytes requested by the PCI initiator have been written, the PTBE[3:0]# are all be deasserted, and the Add-On asserts PTRDY#.

Figure 11 shows Pass-Thru operation for a region defined for an 8-bit Add-On bus interface. As the 8-bit device is connected only to DQ[7:0], the device must access APTD one byte at a time.

The PCI initiator has performed a 32-bit write of 08D49A30h to Pass-Thru region zero. PTBE[3:0]# are all asserted. At clock 1, the Add-On begins reading the APTD Register (asserting SELECT#, ADR[6:2], and RD#). Add-On logic asserts BE0#, and BYTE0 of APTD is driven on DQ[7:0]. At the rising edge of clock 2, BE0# is sampled by the S5933 and PTBE0# is deasserted. PTBE[3:1]# are still asserted.

During clock 2, only BE1# is activated, and BYTE1 of APTD is driven on DQ[7:0]. At the rising edge of clock 3, BE1# is sampled by the S5933 and PTBE1# is deasserted. PTBE[3:2]# are still asserted.

This process continues until all bytes have been read from the APTD Register. During clock 5, RD# is deasserted and PTRDY# is asserted. PTRDY# is sampled by the S5933 at the rising edge of clock 6, and the current data phase is completed. PTATN# is deasserted and new data can be written from the PCI bus. In this example, the byte enables are asserted, sequentially, from BE0# to BE3#. This is not required, bytes may be accessed in any order.

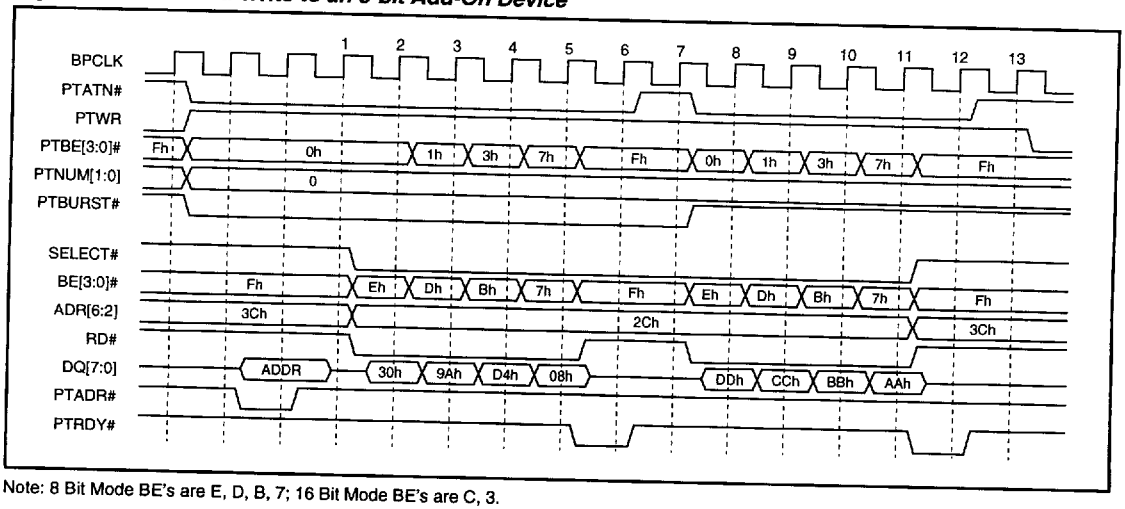
New data is written by the PCI initiator and is available in the APTD Register during clock 7. RD# is asserted and the byte enables are cycled again. With each new data from the PCI bus, the Add-On sequences through the byte enables to access APTD via DQ[7:0].

For 16-bit peripheral devices, the byte steering works in the same way. Because the Add-On data bus is 16-bits wide, only two 16-bit cycles are required to access the entire APTD Register. Two byte enables can be asserted during each access.

In Figure 11, RD# is held low and the byte enables are changed each clock. This assumes the Add-On can accept data at one byte per clock. This is the fastest transfer possible. For slower devices, wait states may be added.

As long as the byte enables remain in a given state, the corresponding byte of the APTD Register is connected to the DQ bus (the RD# or WR# pulse may also be lengthened). Each access may be extended for slower Add-On devices, but extending individual data phases for Pass-Thru cycles may result in the S5933 requesting retries by the initiator.

Figure 11. Pass-Thru Write to an 8-bit Add-On Device



Note: 8 Bit Mode BE's are E, D, B, 7; 16 Bit Mode BE's are C, 3.

CONFIGURATION

The S5933 Pass-Thru interface utilizes four Base Address Registers (BADR1:4). Each Base Address Register corresponds to a Pass-Thru region. The contents of these registers during initialization determine the characteristics of that particular Pass-Thru region. Each region can be mapped to memory or I/O space. Memory mapped devices can, optionally, be mapped below 1 Mbyte and can be identified as prefetchable. Both memory and I/O regions can be configured as 8-, 16-, or 32-bits wide.

The designer has the option to use 1, 2, 3, 4 or none of the Pass-Thru regions. Base Address Registers are loaded during initialization from the external non-volatile boot device. Without an external boot device, the default value for the BADR registers is zero (region disabled). The Base Address Registers are the only registers that define Pass-Thru operation.

S5933 Base Address Register Definition

Some bits in the Base Address Registers have specific functions. The following bits have special functions:

D0 Memory or I/O mapping. If this bit is clear, the region should be memory mapped. If this bit is set, the region should be I/O mapped.

D2:1 Location of a memory region. These bits request that the region be mapped in a particular part of memory. These bit definitions are only used for memory mapped regions.

D2	D1	Location
0	0	Anywhere in 32-bit memory space
0	1	Below 1 Mbyte in memory space (Real Mode address space)
1	0	Anywhere in 64-bit memory space (not valid for the S5933)
1	1	Reserved

D3 Prefetchable. For memory mapped regions, the region can be defined as cacheable. If set, the region is cacheable. If this bit is clear, the region is not.

D31:30 Pass-Thru region bus width. These two bits are used by the S5933 to define the data bus width for a Pass-Thru region. Regardless of the programming of other bits in the BADR register, if D31:30 are zeros, the Pass-Thru region is disabled.

D31	D30	Add-On Bus Width
0	0	Region disabled
0	1	8-bits
1	0	16-bits
1	1	32-bits

BADR1:4 bits D31:30 are used only by the S5933. When the host reads the Base Address Registers during configuration cycles, they always return the same value as D29. If D29 is zero, D31:30 return zero, indicating the region is disabled. If D29 is one, D31:30 return one. This operation limits each Pass-Thru region to a maximum size of 512 Mbytes of memory.

For I/O mapped regions, the PCI specification allows no more than 256 bytes per region. The S5933 allows larger regions to be requested by the Add-On, but a PCI BIOS will not allocate the I/O space and will probably disable the region.

Creating a Pass-Thru Region

Page 3-40 describes the values that must be programmed into the non-volatile boot device to request various block sizes and characteristics for Pass-Thru regions. After reset, the S5933 downloads the contents of the boot device locations 54h, 58h, 5Ch, and 60h into "masks" for the corresponding Base Address Registers. The following are some examples for various Pass-Thru region definitions:

NV Memory Contents	Pass-Thru Region Definition
54h = BFFFF002h	Pass-Thru region 1 is a 4Kbyte region, mapped below 1 Mbyte in memory space with a 16-bit Add-On data bus. This memory region is not cacheable.
58h = 3xxxxxxxh	Pass-Thru region 2 is disabled. (D31:30 = 00.)
60h = FFFFFF81h	Pass-Thru region 3 is a 32-bit, 128 byte I/O-mapped region.
64h = 00000000h	Pass-Thru region 4 is disabled.

During the PCI bus configuration, the host CPU writes all ones to each Base Address Register, and then reads the contents of the registers back. The mask downloaded from the boot device determines which bits are read back as zeros and which are read back as ones. The number of zeros read back indicates the amount of memory or I/O space a particular S5933 Pass-Thru region is requesting.

After the host reads all Base Address Registers in the system (as every PCI device implements from one to six), the PCI BIOS allocates memory and I/O space to each Base Address region. The host then writes the start address of each region back into the Base Address Registers. The start address of a region is always an integer multiple of the region size. For example, a 64 Kbyte memory region is always mapped to begin on a 64K boundary in memory. It is important to note that no PCI device can xbe absolutely located in system memory or I/O space. All mapping is determined by the system, not the application.

Accessing a Pass-Thru Region

After the system is finished defining all Base Address Regions within a system, each Base Address Register contains a physical address. The application software must now find the location in memory or I/O space of its hardware. PCI systems provide BIOS or operating system function calls for application software to find particular devices on the PCI bus based on Vendor ID and Device ID values. This allows application software to access the device's Configuration Registers.

The Base Address Register values in the S5933's Configuration Space may then be read and stored for use by the program to access application hardware. The value in the Base Address Registers is the physical address of the first location of that Pass-Thru region. Some processor architectures allow this address to be used directly to access the PCI device. For Intel Architecture systems, the physical address must be changed into a Segment/Offset combination.

For Real Mode operation in an Intel Architecture system (device mapped below 1 Mbyte in memory), creating a Segment/Offset pair is relatively simple. To calculate a physical address, the CPU shifts the segment register 4 bits to the left and adds the offset (resulting in a 20 bit physical address). The value in the Base Address Register must be read and shifted 4 bits to the right. This is the segment value and should be stored in one of the Segment registers. An offset of zero (stored in SI, DI or another offset register) accesses the first location in the Pass-Thru region.

ABSOLUTE MAXIMUM RATINGS

Parameter	Min	Max	Units
Storage Temperature	–55	125	°C
Supply Voltage (V_{cc})	–0.3	7.0	Volts
Input Pin Voltage	–0.5	V_{cc}+ 5.0	Volts
Power Dissipation		1.05	Watts @ 33 MHz

DC CHARACTERISTICS

The Following table summarizes the required parameters defined by the PCI specification as they apply to the S5933 controller.

PCI Input/Output Electrical Characteristics

Symbol	Parameter	Min	Max	Units	Test Conditions	Notes
V_{CC}	Supply Voltage	4.75	5.25	V		
V_{ih}	Input High Voltage	2.0		V		
V_{il}	Input Low Voltage	–0.5	0.8	V		
I_{ih}	Input High Leakage Current		70	µA	V_{in} = 2.7	1
I_{il}	Input Low Leakage Current		–70	µA	V_{in} = 0.5	1
V_{oh}	Output High Voltage	2.4		V	I_{out} = –2mA	
V_{ol}	Output Low Voltage		0.55	V	I_{out} = 3mA, 6mA	2
C_{in}	Input Pin Capacitance		10	pF		3
C_{clk}	CLK Pin Capacitance	5	12	pF		
C_{IDSEL}	IDSEL Pin Capaticance		8	pF		

Notes:
1. Input leakage applies to all inputs and bi-directional buffers.
2. PCI Bus signals without pull-up resistors will provide the 3 mA output current. Signals which require a pull-up resistor will provide 6 mA output current.
3. The PCI specification limits all PCI inputs not located on the motherboard to 10 pf (the clock is allowed to be 12 pf).

PCI Bus Signals

The following table summarizes the PCI Bus DC parameters defined by the PCI specification as they apply to the S5933 controller.

Signal	Type	Direction	Max	Units	Notes
CLK		Input			
RST#		Input			
INTA#	Open Drain	Output	4	mA	
AD[31:0]	t/s	Bi-directional		mA	
REQ#	t/s	Output	4	mA	
GNT#		Input			
C/BE[3:0]#	t/s	Bi-directional	4	mA	
DEVSEL#	s/t/s	Bi-directional		mA	
FRAME#	s/t/s	Bi-directional	4	mA	
IRDY#	s/t/s	Bi-directional	4	mA	
TRDY#	s/t/s	Bi-directional	4	mA	
PERR#	s/t/s	Bi-directional	4	mA	
PAR	t/s	Bi-directional	4	mA	
SERR#	Open Drain	Output	4	mA	
STOP#	s/t/s	Bi-directional	4	mA	
LOCK#		Input			
IDSEL		Input			

Add-On Bus Signals

The following table summarizes the Add-On Bus DC parameters as they apply to the S5933 controller.

Signal	Type	Direction	Max	Units	Notes
PCLK		Output	8	mA	
IRQ#		Output	4	mA	
SYSRST#		Output	4	mA	
ADR[6:2]		Input			
SELECT		Input			
ADR[6:2]		Input			
BE[3:0]#		Input			
RD#		Input			
WR#		Input			
DQ[31:0]	t/s	Bi-directional	4	mA	
WRFULL		Output	4	mA	
RDEMPTY		Output	4	mA	
RDFIFO#		Input			
WRFIFO#		Input			
PTATN#		Output	4	mA	
PTBURST#		Output	4	mA	
PTADR#		Input			
PTRDY#		Input			
PTWR		Output	4	mA	
PTBE[3:0]#		Output	4	mA	
PTNUM[1:0]		Output	4	mA	
EQ[7:0]	t/s	Bi-directional	1	mA	
EA[8:0]	t/s	Output	1	mA	
EA[15:9]		Output	1	mA	
MODE		Input			
TEST		Output	4	mA	
FLT#		Input			
ERD#/SCL		Output	1	mA	
EWR#/SDA	t/s	Bi-directional	1	mA	

3

AC CHARACTERISTICS

PCI Bus Timings

Functional Operation Range (V_{CC}=5.0V ±5%, 0°C to 70°C, 50 pF load on outputs)

Symbol	Parameter	Min	Max	Units	Notes
TCL	Cycle Time	30		ns	
t1	High Time	12		ns	
t2	Low Time	12		ns	
t3	Rise Time (0.8V to 2.0V)		3	ns	
t4	Fall Time (2.0V to 0.8V)		3	ns	
t5	Output Valid Delay (Bussed Signals) Output Valid Delay (Point-to-Point Signals)	2 2	11 12	ns	Note 1
t6	Float to Active Delay	2		ns	
t7	Active to Float Delay		28	ns	
t8	Rising Edge Setup (Bussed Signals) Rising Edge Setup (GNT#) Rising Edge Setup (REQ#)	7 10 12		ns	
t9	Hold from PCI Clock Rising Edge	0		ns	
t10	PCICLK to BPCLK Delay	2	6.5	ns	

Notes:
1. Minimum times are for unloaded outputs, maximum times are for 50 pF equivalent loads.

Figure 1. PCI Clock Timing

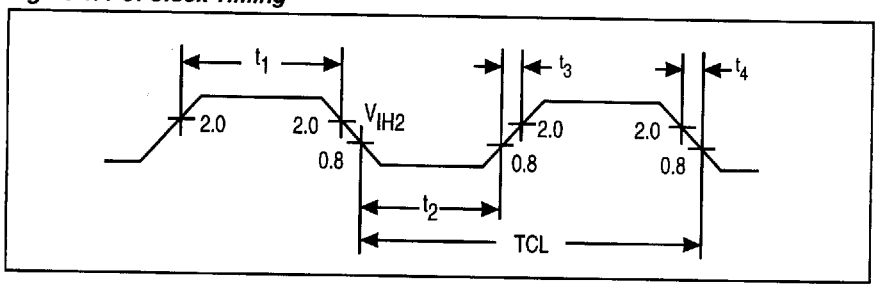


Figure 2. PCI Output Timing

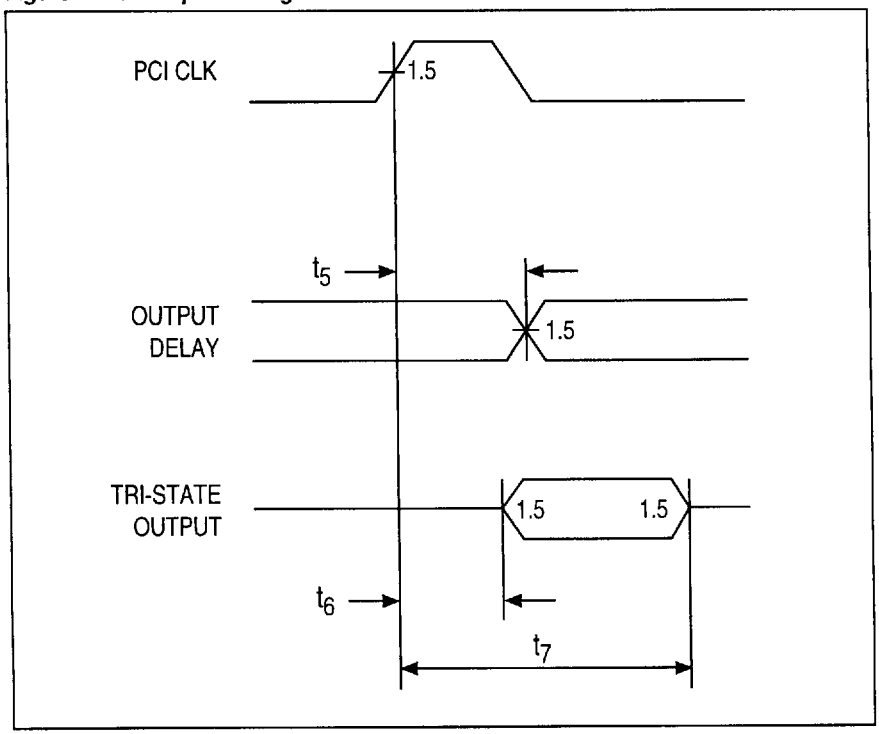


Figure 3. PCI Input Timing

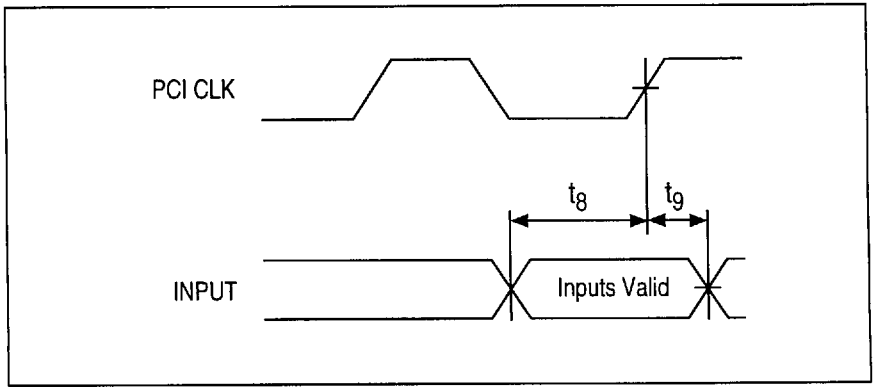


Figure 4. Add-On Clock Timing

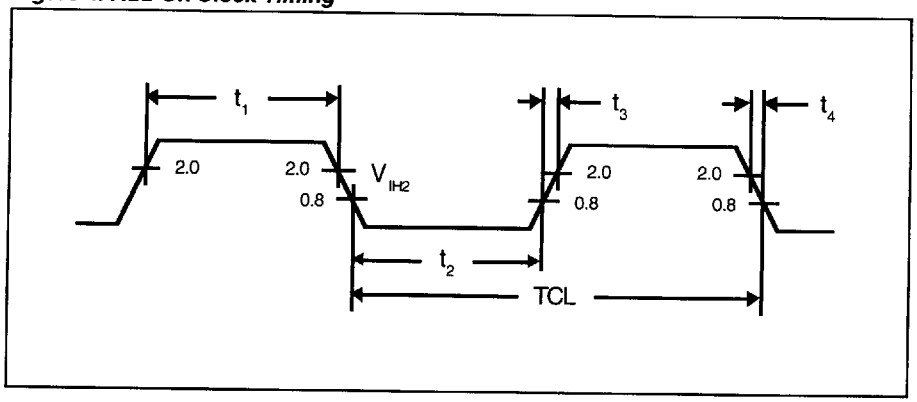
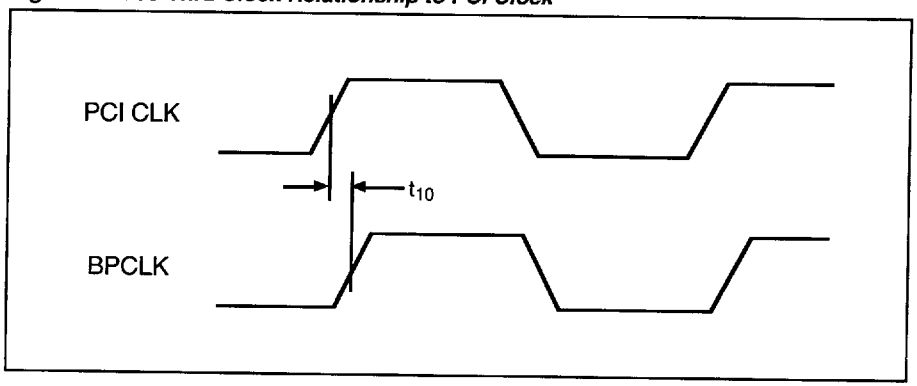


Figure 5. Pass-Thru Clock Relationship to PCI Clock



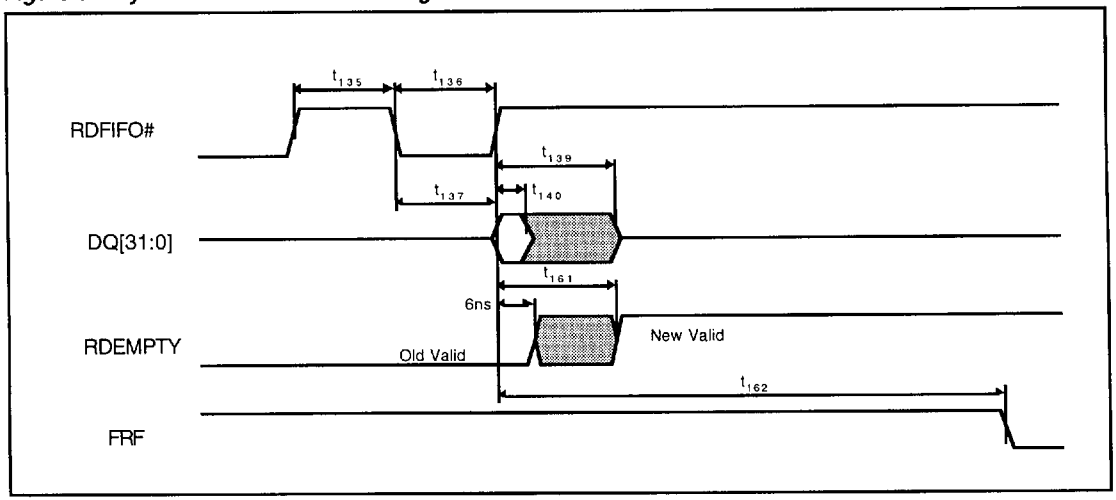
Asynchronous RDFIFO# Timing

Functional Operation Range (V_{CC}= 5.0V 5%, 0°C to 70°C T_a, 50 pf load on outputs).

Symbol	Parameter	Min	Max	Units	Notes
t_{135}	RDFIFO# High Time	17		ns	
t_{136}	RDFIFO# Low Time	17		ns	
t_{137}	RDFIFO# Low to DQ[31:0] Driven		21	ns	
t_{139}	RDFIFO# High to DQ[31:0] Float		20	ns	
t_{140}	DQ[31:0] Hold from RDFIFO# Rising Edge	5		ns	
t_{161}	PCI to ADD-ON FIFO RDEMPTY Valid from RDFIFO# Rising Edge		15	ns	
t_{162}	PCI to ADD-ON FIFO FRF Valid from RDFIFO# Rising Edge		85	ns	

3

Figure 6. Asynchronous RDFIFO# Timing

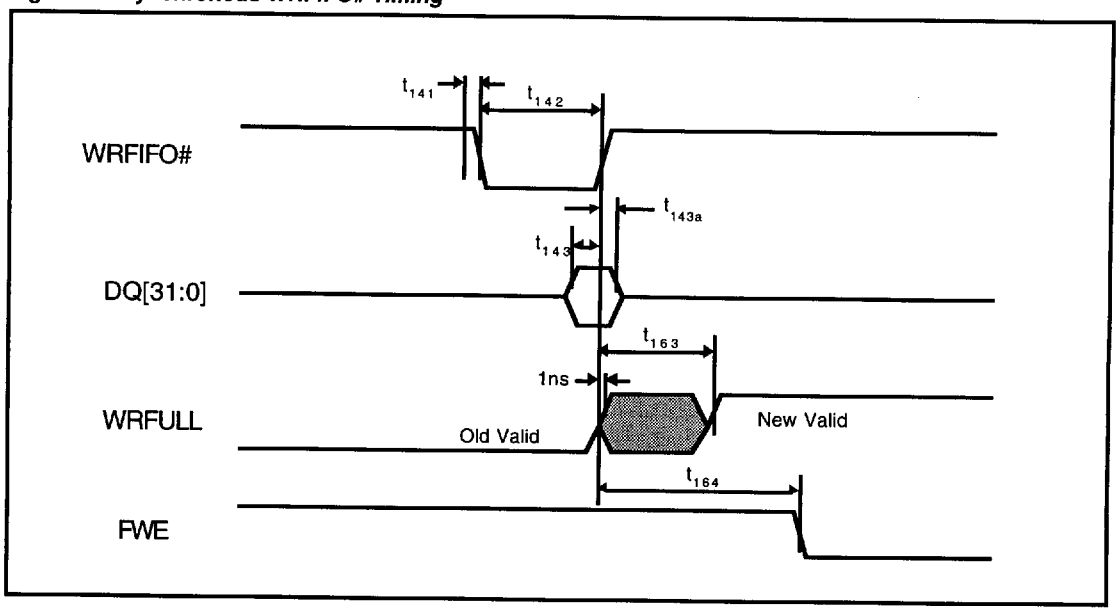


Asynchronous WRFIFO# Timing

Functional Operation Range (V_{CC}=5.0V 5%, 0°C to 70°C T_a, 50 pf load on outputs).

Symbol	Parameter	Min	Max	Units	Notes
t_{141}	WRFIFO# High Time	2		ns	
t_{142}	WRFIFO# Low Time	17		ns	
t_{143}	DQ[31:0] Setup to WRFIFO# Rising Edge	4		ns	
t_{143a}	DQ[31:0] Hold from WRFIFO# Rising Edge	2		ns	
t_{163}	ADD-ON to PCI FIFO WRFULL Valid from WRFIFO# Rising Edge		16	ns	
t_{164}	ADD-ON to PCI FIFO FWE Valid from WRFIFO# Rising Edge		28	ns	

Figure 7. Asynchronous WRFIFO# Timing



Synchronous RDFIFO# Timing

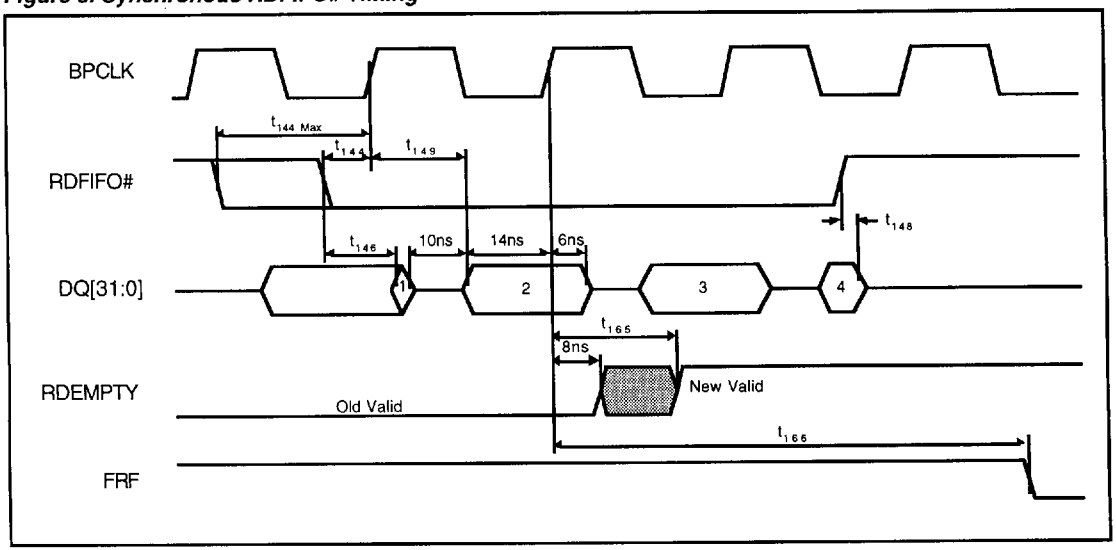
Functional Operation Range (V_{CC}=5.0V 5%, 0°C to 70°C T_a, 50 pf loaf on outputs).

Symbol	Parameter	Min	Max	Units	Notes
t_{144}	RDFIFO# Setup tp BPCLK Rising Edge	8	26	ns	1
t_{145}	RDFIFO# Low Time	8		ns	
t_{146}	RDFIFO# Low to DQ[31:0] Driven		12	ns	
t_{148}	RDFIFO# High to DQ[31:0] Float		3	ns	
t_{149}	DQ[31:0] Valid from BPCLK Rising Edge		16	ns	3
t_{165}	PCI to ADD-ON FIFO RDEMPTY Valid from BPCLK Rising Edge		12	ns	2
t_{166}	PCI to ADD-ON FIFO FRF Valid from BPCLK Rising Edge		80	ns	

Notes:
1. Min and Max times are indicated to allow increased valid data time as shown by dashed lines.
2. State change of RDEMPTY shown below is reference only. Actual change would indicate no Data 3 available.
3. Valid applies after first access. First access is async with following as sync accesses.

3

Figure 8. Synchronous RDFIFO# Timing



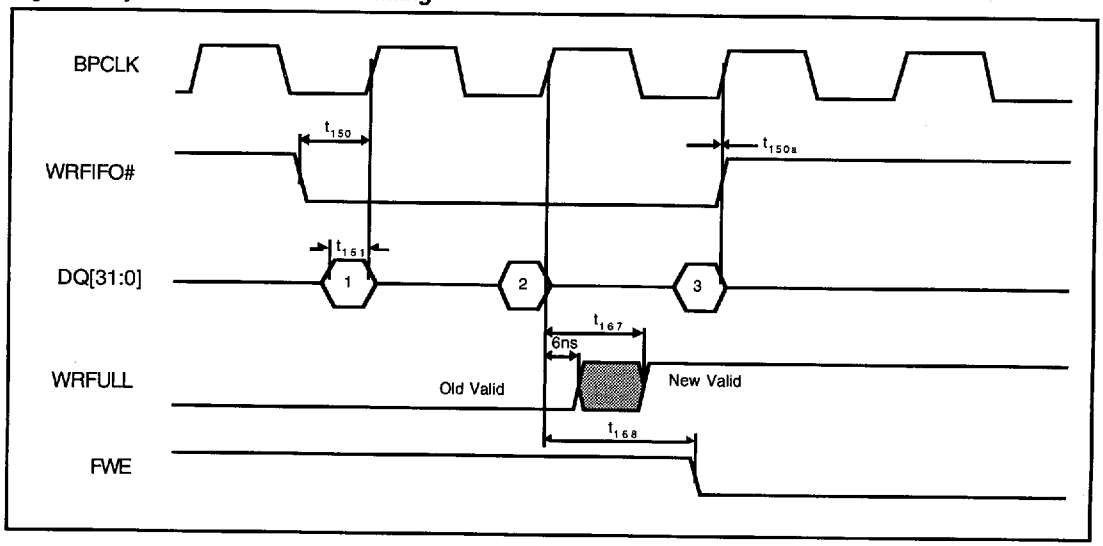
Synchronous WRFIFO# Timing

Functional Operation Range (V_{CC}= 5.0V 5%, 0°C to 70°C T_a, 50 pf load on outputs).

Symbol	Parameter	Min	Max	Units	Notes
t_{150}	WRFIFO# Setup to BPCLK Rising Edge	12		ns	
t_{150a}	WRFIFO# Hold Time to BPCLK Rising Edge		0	ns	
t_{151}	DQ[31:0] Setup to BPCLK Rising Edge	7			
t_{151a}	DQ[31:0] Hold from BPCLK Rising Edge		0		
t_{167}	ADD-ON to PCI WRFULL Valid from BPCLK Rising Edge		11	ns	1
t_{168}	ADD-ON to PCI FIFO FWE Valid from BPCLK Rising Edge		26	ns	

Notes:
1. State change of WRFULL shown below is reference only. Actual change would indicate no Data 3 written.

Figure 9. Synchronous WRFIFO# Timing



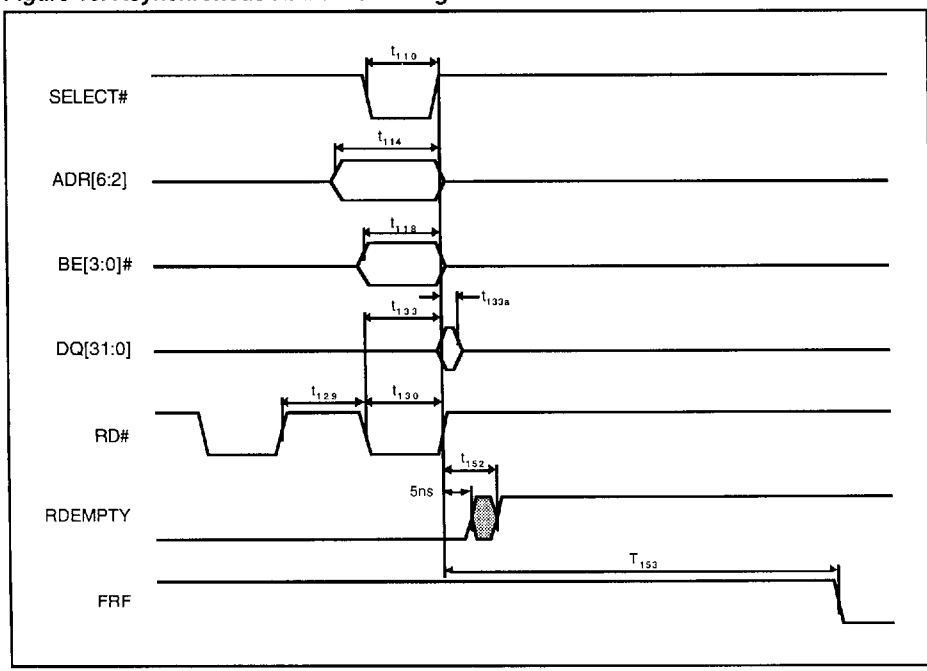
Asynchronous RD# FIFO and Register Access Timing

Functional Operation Range (V_{CC}=5.0V 5%, 0°C to 70°C T_a, 50 pf load on outputs).

Symbol	Parameter	Min	Max	Units	Notes
t_{110}	SELECT# Setup to RD# Rising Edge	10		ns	
t_{114a}	SELECT# Hold from RD# Rising Edge	-1		ns	
t_{114}	ADR[6:2] Setup to RD# Rising Edge	18		ns	
t_{114a}	ADR[6:2] Hold from RD# Rising Edge	0			
t_{118}	BE[3:0]# Setup to RD# Rising Edge	12		ns	
t_{118a}	BE[3:0]# Hold from RD# Rising Edge	0		ns	
t_{129}	RD# High Time	16		ns	
t_{130}	RD# Low Time	15		ns	
t_{133}	DQ[31:0] Valid from RD# Falling Edge	15		ns	
t_{133a}	DQ[31:0] Hold from RD# Rising Edge	3		ns	
t_{152}	RDEMPTY Status Valid from RD# Rising Edge		10	ns	
t_{153}	FRF Status Valid from RD# Rising Edge		75	ns	

3

Figure 10. Asynchronous RD# FIFO Timing

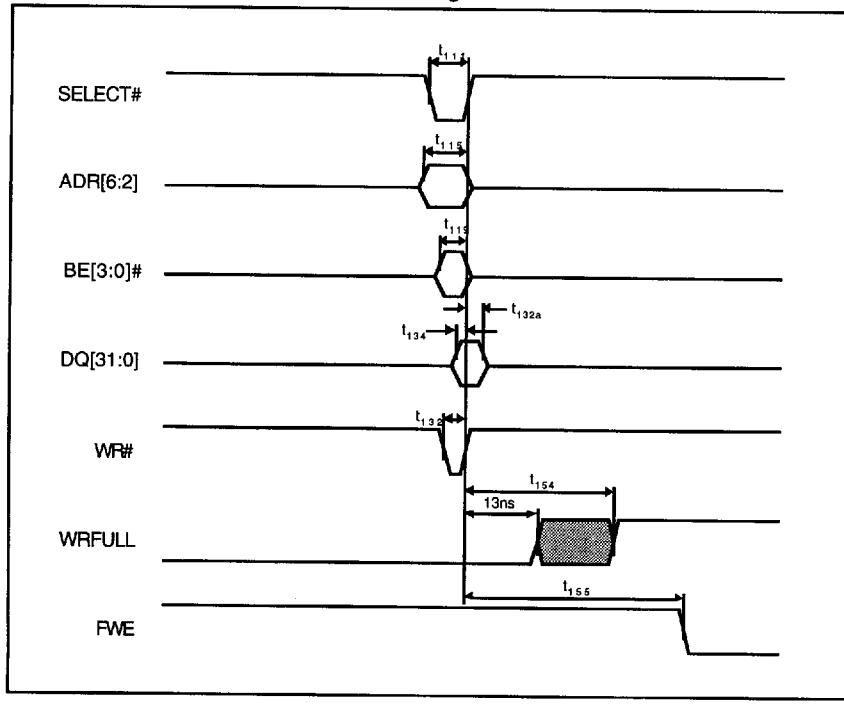


Asynchronous WR# FIFO and Register Access Timing

Functional Operation Range (V_{CC}=5.0V 5%, 0°C to 70°C T 50 pf load on outputs).

Symbol	Parameter	Min	Max	Units	Notes
t_{111}	SELECT# Setup to WR# Rising Edge	7		ns	
t_{111a}	SELECT# Hold from WR# Rising Edge	0		ns	
t_{115}	ADR[6:2] Setup to WR# Rising Edge	8		ns	
t_{115a}	ADR[6:2] Hold from WR# Rising Edge	0		ns	
t_{119}	BE[3:0]# Setup to WR# Rising Edge	5		ns	
t_{119a}	BE[3:0]# Hold from WR# Rising Edge	0		ns	
t_{131}	WR# High Time	TBD		ns	
t_{132}	WR# Low Time	4		ns	
t_{134}	DQ[31:0] Setup to WR# Rising Edge	2		ns	
t_{134a}	DQ[31:0] Hold from WR# Rising Edge	3		ns	
t_{154}	WRFULL Status Valid from WR# Rising Edge		27	ns	
t_{155}	FWE Status Valid from WR# Rising Edge		40	ns	

Figure 11. Asynchronous WR# FIFO Timing



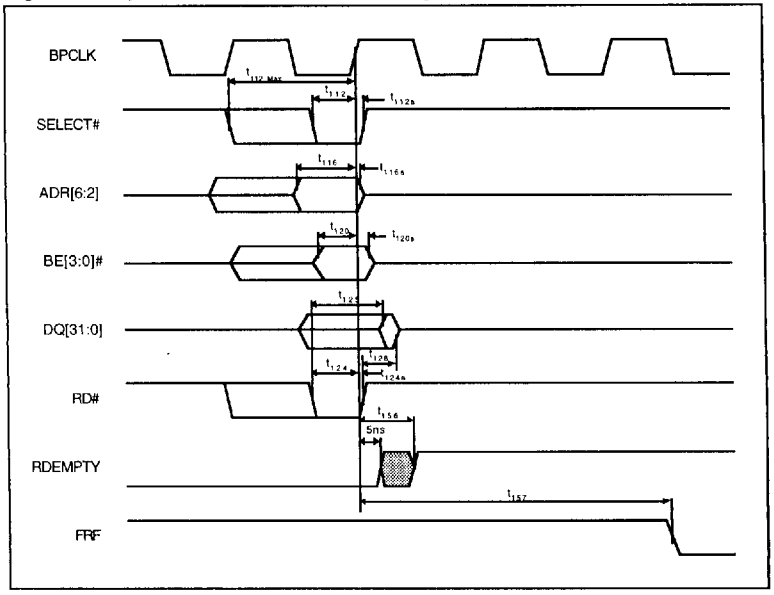
Synchronous RD# FIFO Timing

Functional Operation Range (V_{CC}=5.0V 5%, 0°C to 70°C T 50 pf load on outputs).

Symbol	Parameter	Min	Max	Units	Notes
t_{112}	SELECT# Setup to BPCLK Rising Edge	10	30	ns	4
t_{112a}	SELECT# Hold from BPCLK Rising Edge	2		ns	
t_{116}	ADR[6:2] Setup to BPCLK Rising Edge	14	34	ns	4
t_{116a}	ADR[6:2] Hold from BPCLK Rising Edge	1		ns	
t_{120}	BE[3:0]# Setup to BPCLK Rising Edge	9	29	ns	4
t_{120a}	BE[3:0]# Hold from BPCLK Rising Edge	3		ns	
t_{125}	RD# Low to DQ[31:0] Driven		17	ns	1
t_{128}	RD# High to DQ[31:0] Float		8	ns	
t_{156}	RDEMPTY Status Valid to BPCLK Rising Edge		13	ns	
t_{157}	FRF Status Valid to BPCLK Rising Edge		74	ns	
t_{124}	RD# Setup to BPCLK Rising Edge	11	31	ns	4
t_{124a}	RD# Hold from BPCLK Rising Edge	1		ns	
t_{127}	DQ[31:0] Valid from BPCLK Rising Edge		6	ns	

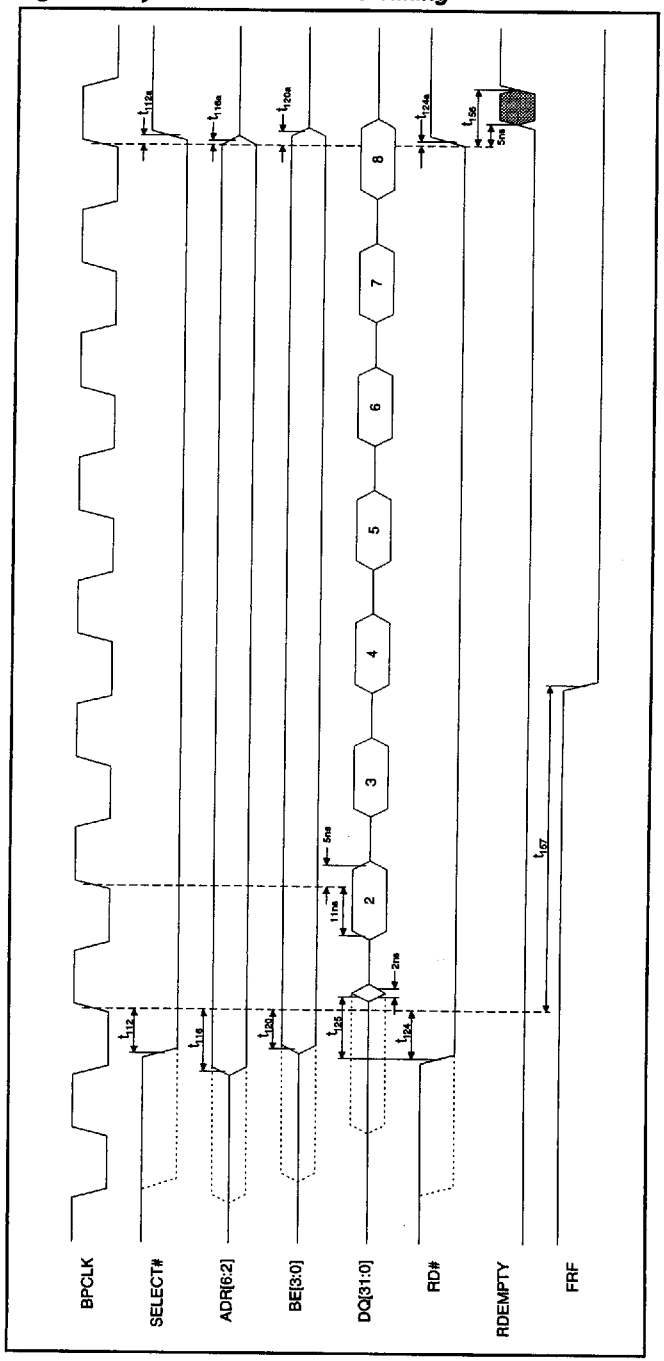
Notes:
1. Data is valid for 22ns for a 31ns t_{124} RD# Setup.
2. RD# and SELECT# must both be asserted to dric=ve DQ[31:0] - delay is from the last one asserted.
3. When increasing Setup times, ADR[6:2], BE[3:0]#, SELECT#, and RD# timing relations remain relative to each other as shown.
4. Min and Max are indicated to allow increased valid data time as shown by dashed lines. First accesses are async.

Figure 12. Synchronous RD# FIFO Timing



Synchronous Multiple RD# FIFO Timing

Figure 13. Synchronous RD# FIFO Timing



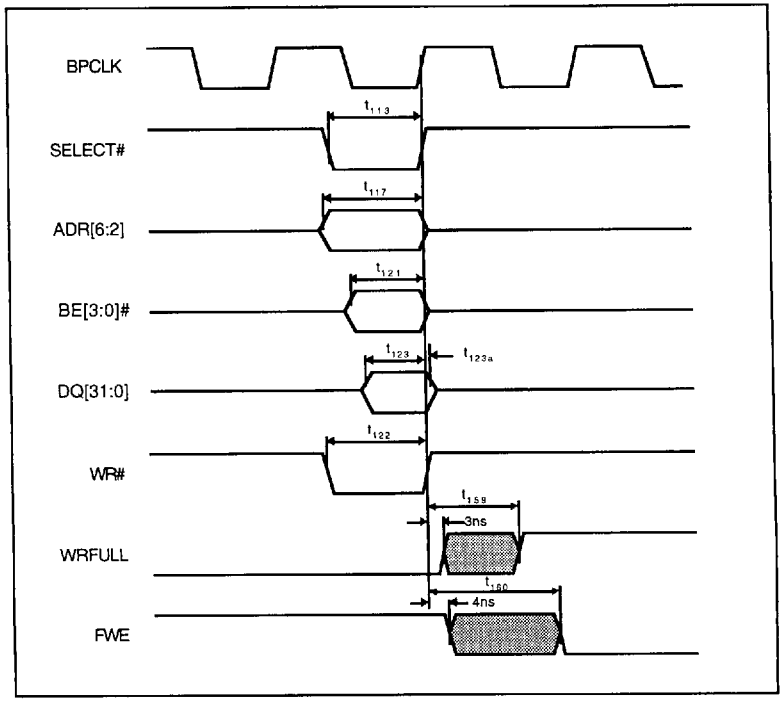
Synchronous WR# FIFO Timing

Functional Operation Range (V_{CC}=5.0V 5%, 0°C to 70°C T_a, 50 pf load on outputs).

Symbol	Parameter	Min	Max	Units	Notes
t_{113}	SELECT# Setup to BPCLK Rising Edge	19		ns	
t_{113a}	SELECT# Hold from BPCLK Rising Edge	0		ns	
t_{117}	ADR[6:2] Setup to BPCLK Rising Edge	20		ns	
t_{117a}	ADR[6:2] Hold from BPCLK Rising Edge	0		ns	
t_{121}	BE[3:0]# Setup to BPCLK Rising Edge	15		ns	
t_{121a}	BE[3:0]# Hold from BPCLK Rising Edge	0		ns	
t_{123}	DQ[31:0] Setup to BPCLK Rising Edge	12		ns	
t_{123a}	DQ[31:0] Hold from BPCLK Rising Edge	1		ns	
t_{122}	WR# Setup to BPCLK Rising Edge	20		ns	
t_{122a}	WR# Hold from BPCLK Rising Edge	0		ns	
t_{159}	WRFULL Status Valid to BPCLK Rising Edge		18	ns	
t_{160}	FWE Status Valid to BPCLK Rising Edge		26	ns	

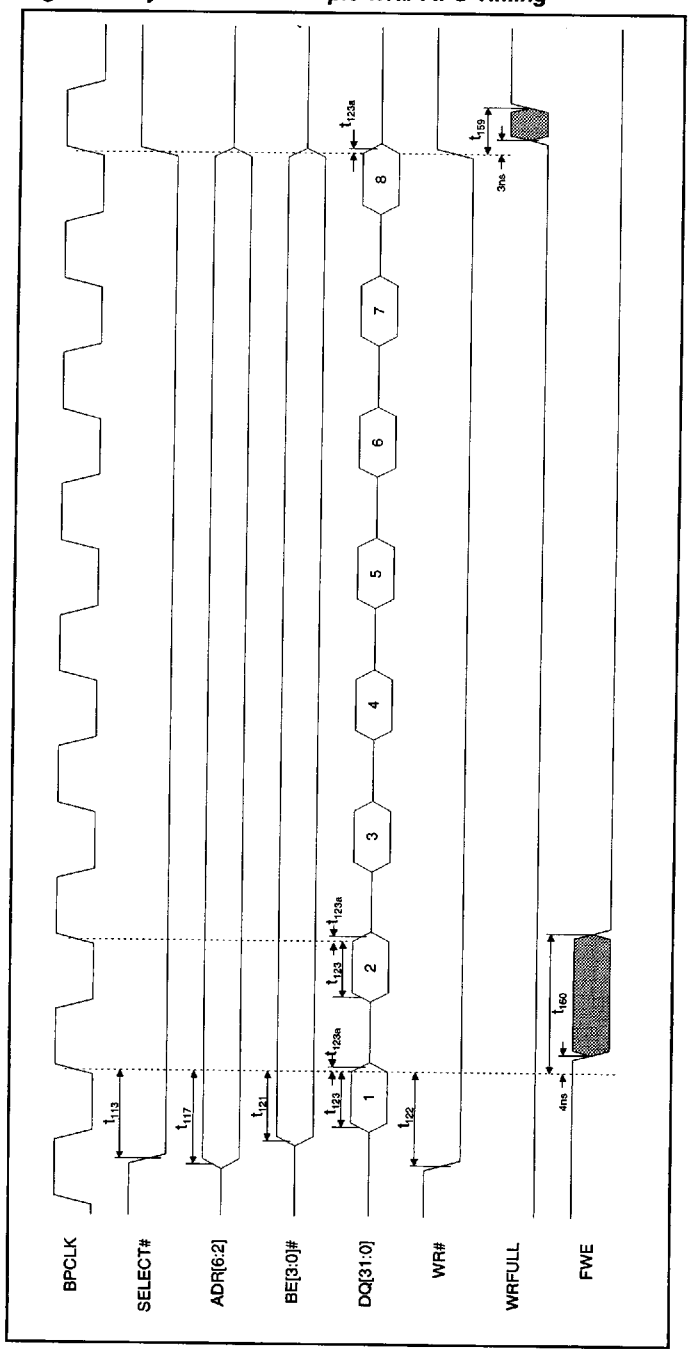
3

Figure 14. Synchronous WR# FIFO Timing



Synchronous Multiple WR# FIFO Timing

Figure 15. Synchronous Multiple WR# FIFO Timing



Target S5933 Pass-Thru Interface Timings

Functional Operation Range (V_{CC}=5.0V ±5%, 0°C to 70°C, 50 pF load on outputs)

Symbol	Parameter	Min	Max	Units	Notes
t_{10a}	SELECT# Setup to BPCLK Rising Edge	3		ns	
t_{11a}	SELECT# Hold from BPCLK Rising Edge	2		ns	
t_{13}	ADR[6:2], BE[3:0]# Setup to BPCLK Rising Edge	5		ns	
t_{14}	ADR[6:2], BE[3:0]# Hold from BPCLK Rising Edge	2		ns	
t_{17}	RD# Low to DQ{31:0] Driven		13	ns	1
t_{24}	Pass-Thru Status Valid from BPCLK Rising Edge		5	ns	
t_{25}	Pass-Thru Status Hold from BPCLK Rising Edge	0		ns	
t_{26}	PTRDY# Setup to BPCLK Rising Edge	5		ns	
t_{27}	PTRDY# Hold from BPCLK Rising Edge	3		ns	
t_{28}	PCICLK to BPCLK delay	2	6.5	ns	
t_{29}	RD#, WR# Setup to BPCLK Rising Edge	5		ns	
t_{30}	RD#, WR# Hold from BPCLK Rising Edge	2		ns	
t_{31}	DQ[31:0] Setup to BPCLK Rising Edge	5		ns	
t_{32}	DQ[31:0] Hold from BPCLK Rising Edge	2		ns	
t_{33}	DQ[31:0] Valid from BPCLK Rising Edge		15	ns	
t_{34}	DQ[31:0] Float from RD# Rising Edge		12	ns	

Notes:
1. This timing also applies to the use of BE[3:0]# to control DQ[31:0] drive.

Figure 17. Pass-Thru Data Register Read Timing

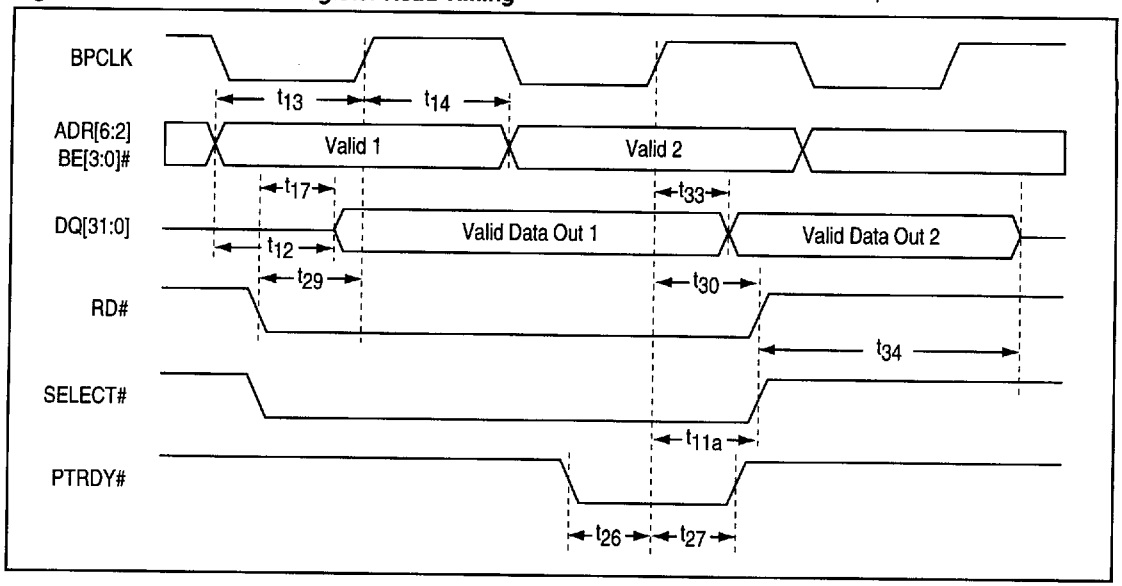


Figure 18. Pass-Thru Data Register Write Timing

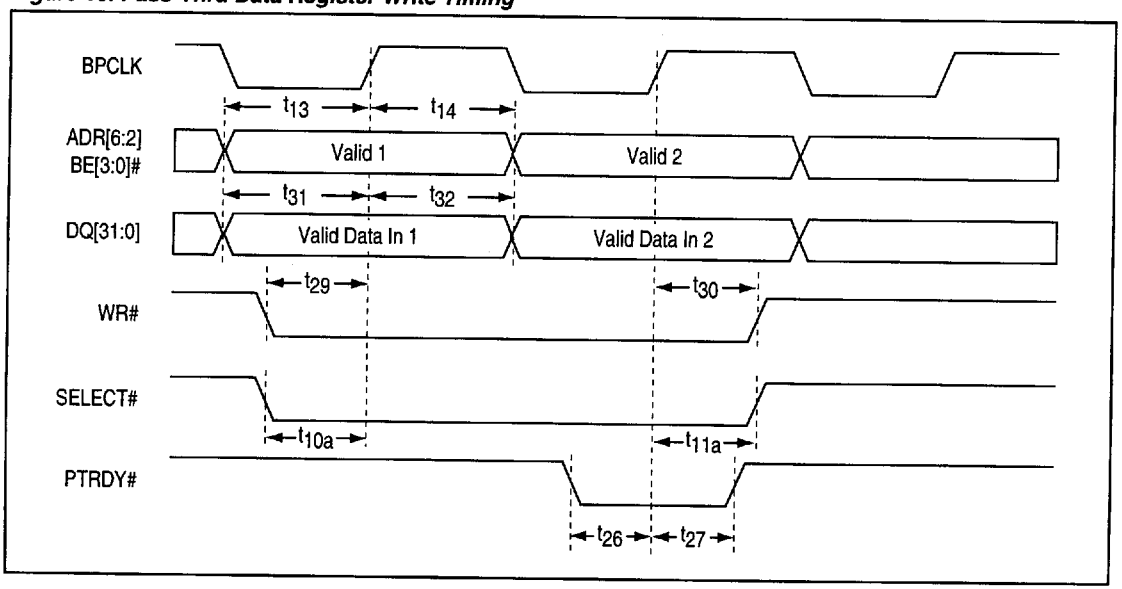
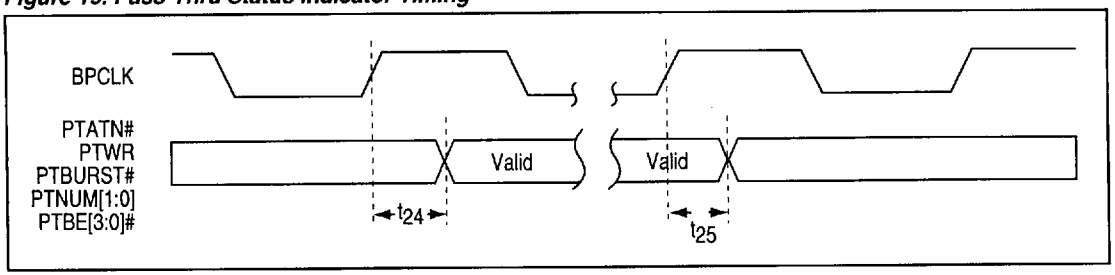


Figure 19. Pass-Thru Status Indicator Timing



Target Byte-Wide nv Memory Interface Timings

Functional Operation Range (V_{CC}=5.0V ±5%, 0°C to 70°C, 50 pF load on outputs)

Symbol	Parameter	Min	Max	Units	Notes
t35	ERD# Cycle Time	8T		ns	Note 1
t36	ERD# Low Time	6T		ns	Note 1
t37	ERD# High Time	2T		ns	Note 1
t38	EA[15:0] Setup to ERD# or EWR# Low	T		ns	Note 1
t39	EA[15:0] Hold from ERD# or EWR# High	T		ns	Note 1
t40	EQ[7:0] Setup to ERD# Rising Edge	10		ns	Note 1
t41	EQ[7:0] Hold from ERD# Rising Edge	2		ns	Note 1
t42	EWR# Cycle Time			ns	Note 1,2
t43	EWR# Low Time	6T		ns	Note 1
t44	EWR# High Time	2T		ns	Note 1
t45	EQ[7:0] Setup to EWR# Low	-10	0	ns	Note 1
t46	EQ[7:0] Hold from EWR# High	T		ns	Note 1

Notes:

1. T represents the clock period for the PCI bus clock (30ns @ 33 MHz).
2. The write cycle time is controlled by both the PCI bus clock and software operations to initiate the write operation of nv memory. This parameter is the result of several software operations to the Bus Master Control/Status Register (MCSR).

Figure 20. nv Memory Read Timing

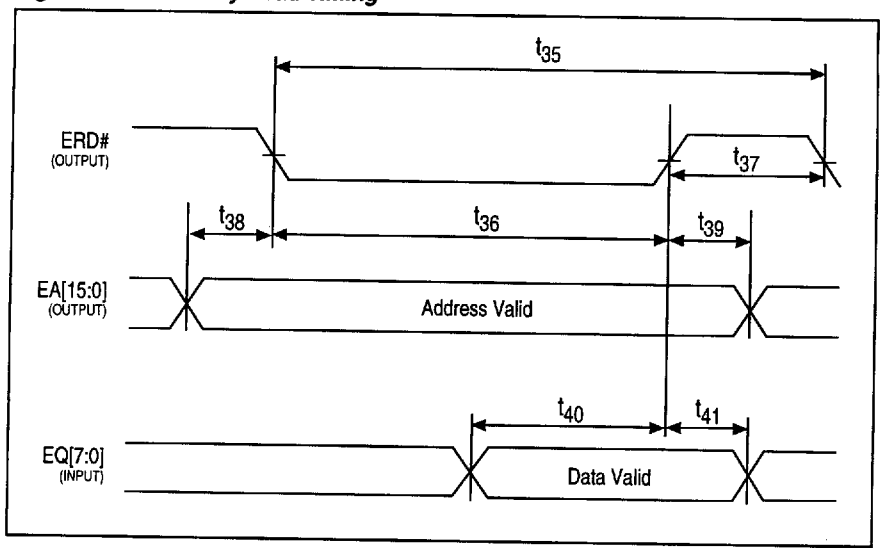
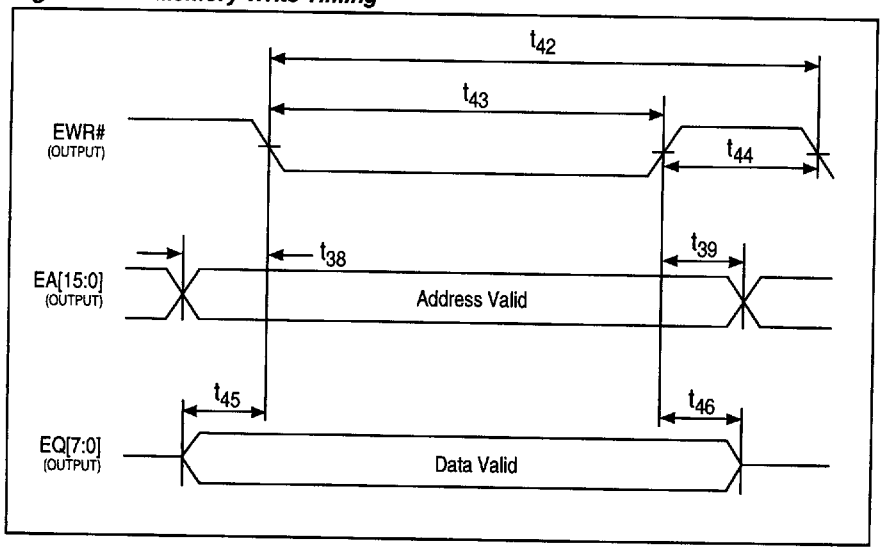


Figure 21. nv Memory Write Timing



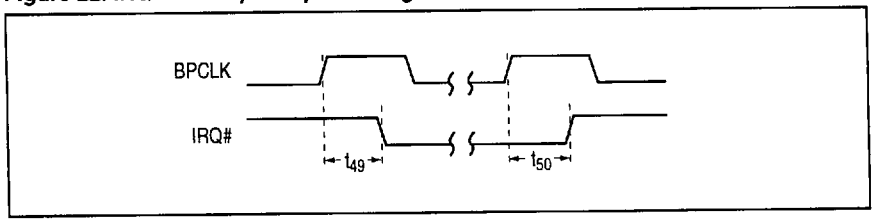
Target Interrupt Timings

Functional Operation Range (V_{CC}=5.0V ±5%, 0°C to 70°C, 50 pF load on outputs)

Symbol	Parameter	Min	Max	Units	Notes
t49	IRQ# Low from BPCLK Rising Edge		15	ns	Note 1
t50	IRQ# High from BPCLK Rising Edge		15	ns	Note 1

Notes:
1. This timing applies to interrupts generated and cleared from the PCI interface.

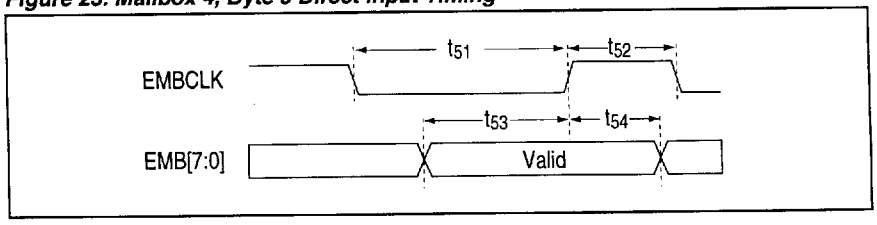
Figure 22. IRQ# Interrupt Output Timing



Functional Operation Range (V_{CC}=5.0V ±5%, 0°C to 70°C, 50 pF load on outputs)

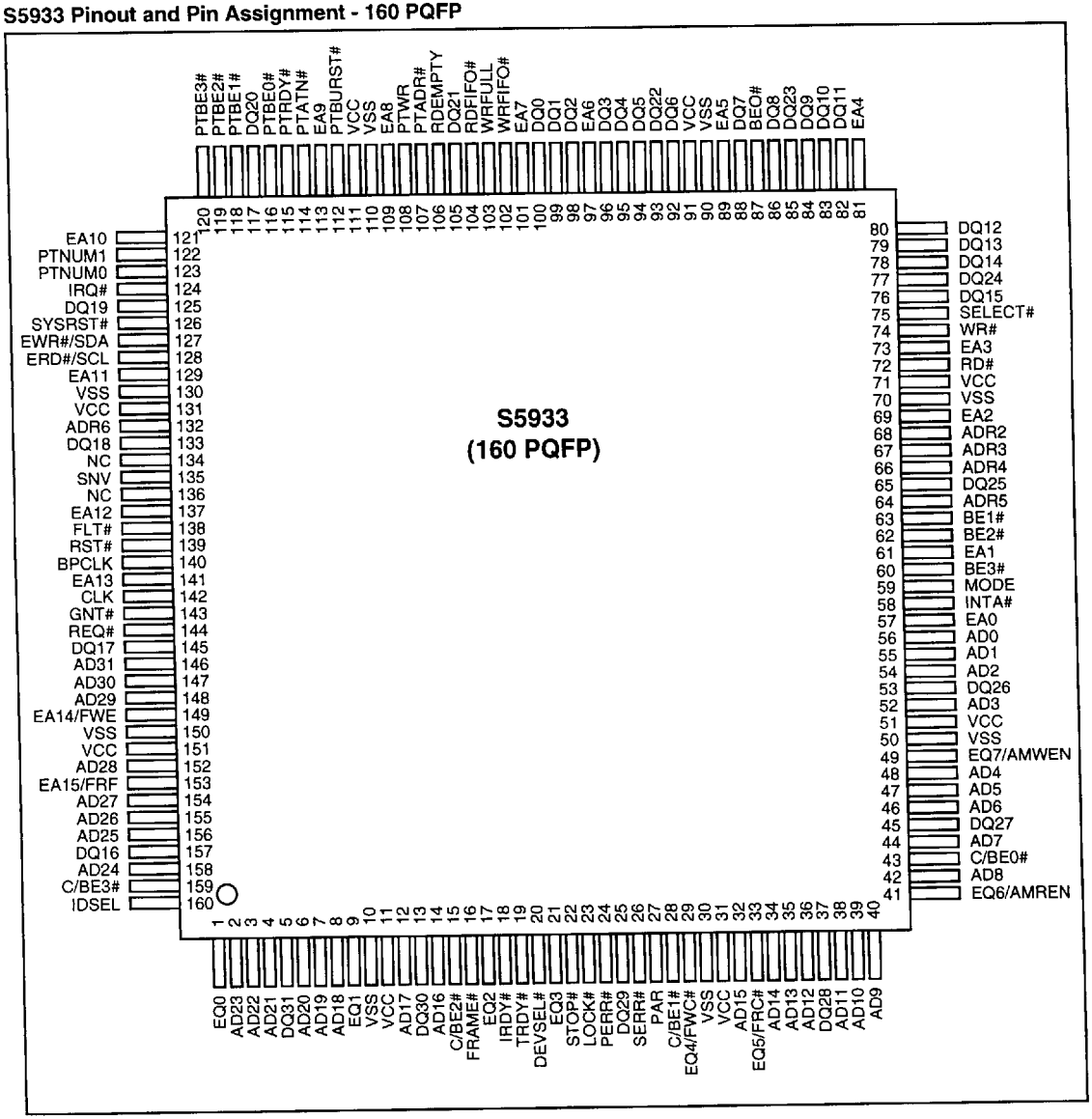
Symbol	Parameter	Min	Max	Units	Notes
t51	EMBCLK Low Time	12		ns	
t52	EMBLK High Time	12		ns	
t53	EMB[7:0] Setup to EMBCLK Rising Edge	5		ns	
t54	EMB[7:0] Hold from EMBCLK Rising Edge	2		ns	

Figure 23. Mailbox 4, Byte 3 Direct Input Timing



PAGE(S) INTENTIONALLY BLANK

S5933 Pinout and Pin Assignment - 160 PQFP



S5933 Pinout and Pin Assignment - 208 TQFP

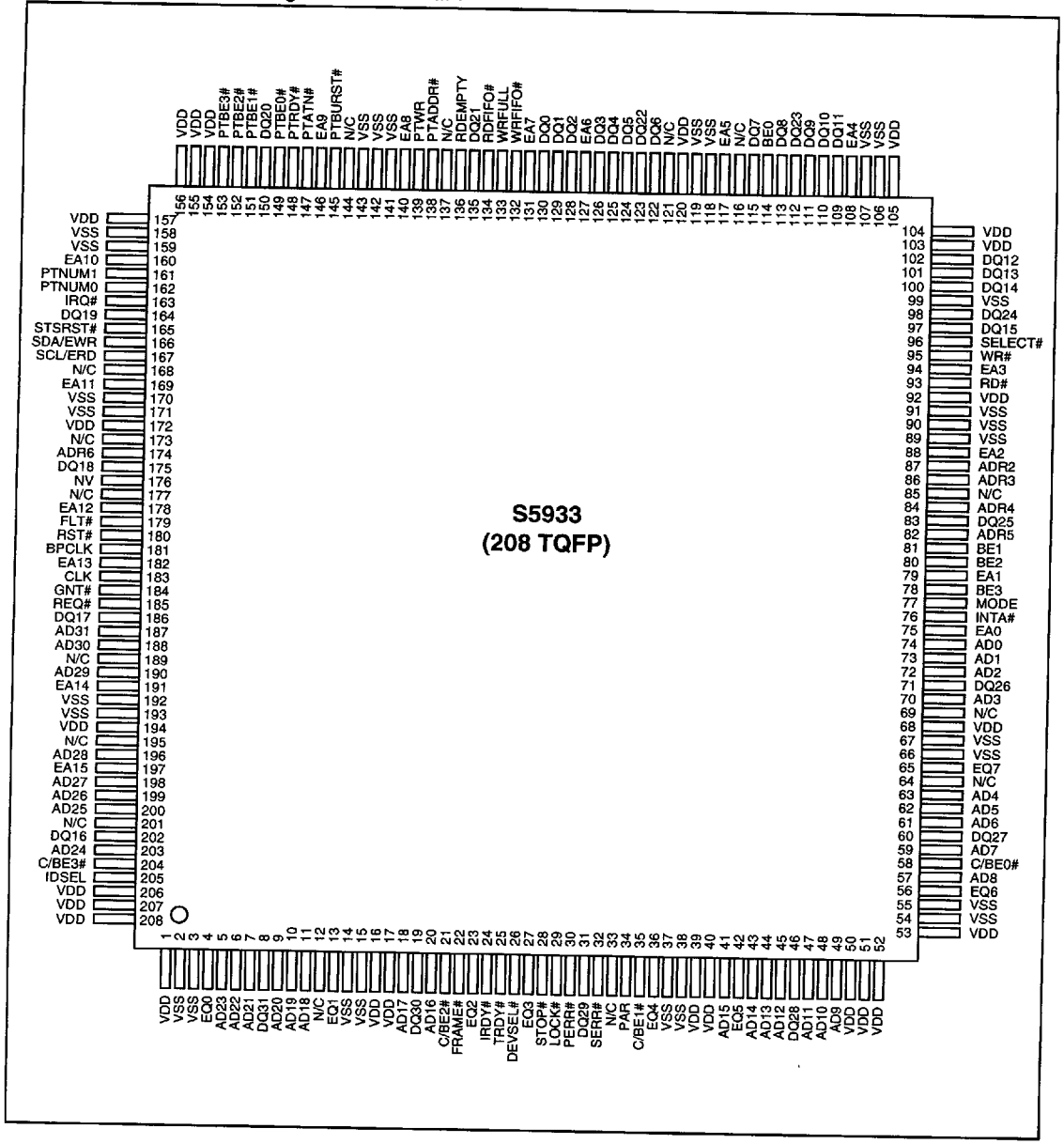


Table 1. S5933 Numerical Pin Assignment - 160 PQFP

Pin#	Signal	Type
1	EQ0	t/s
2	AD23	t/s
3	AD22	t/s
4	AD21	t/s
5	DQ31	t/s
6	AD20	t/s
7	AD19	t/s
8	AD18	t/s
9	EQ1	t/s
10	VSS	V
11	VCC	V
12	AD17	t/s
13	DQ30	t/s
14	AD16	t/s
15	C/BE2#	t/s
16	FRAME#	t/s
17	EQ2	t/s
18	IRDY#	t/s
19	TRDY#	t/s
20	DEVSEL#	t/s
21	EQ3	t/s
22	STOP#	t/s
23	LOCK#	in
24	PERR#	t/s
25	DQ29	t/s
26	SERR#	o/d
27	PAR	t/s
28	C/BE1#	t/s
29	EQ4/FWC#	t/s
30	VSS	V
31	VCC	V
32	AD15	t/s

Pin#	Signal	Type
33	EQ5/FRC#	t/s
34	AD14	t/s
35	AD13	t/s
36	AD12	t/s
37	DQ28	t/s
38	AD11	t/s
39	AD10	t/s
40	AD9	t/s
41	EQ6/AMREN	t/s
42	AD8	t/s
43	C/BE0#	t/s
44	AD7	t/s
45	DQ27	t/s
46	AD6	t/s
47	AD5	t/s
48	AD4	t/s
49	EQ7/AMWEN	t/s
50	VSS	V
51	VCC	V
52	AD3	t/s
53	DQ26	t/s
54	AD2	t/s
55	AD1	t/s
56	AD0	t/s
57	EA0	t/s
58	INTA#	o/d
59	MODE	in
60	BE3#	in
61	EA1	t/s
62	BE2#	in
63	BE1#	in
64	ADR5	in

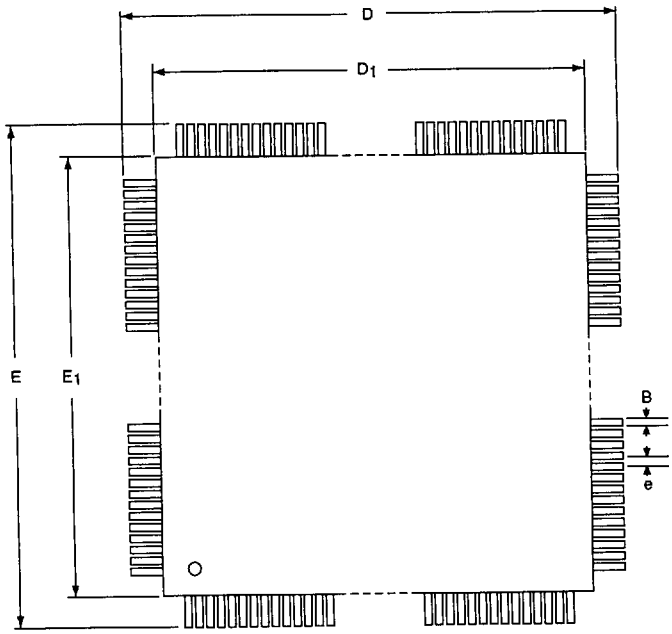
Pin#	Signal	Type
65	DQ25	t/s
66	ADR4	in
67	ADR3	in
68	ADR2	in
69	EA2	t/s
70	VSS	V
71	VCC	V
72	RD#	in
73	EA3	t/s
74	WR#	in
75	SELECT#	in
76	DQ15	t/s
77	DQ24	t/s
78	DQ14	t/s
79	DQ13	t/s
80	DQ12	t/s
81	EA4	t/s
82	DQ11	t/s
83	DQ10	t/s
84	DQ9	t/s
85	DQ23	t/s
86	DQ8	t/s
87	BE0#	in
88	DQ7	t/s
89	EA5	t/s
90	VSS	V
91	VCC	V
92	DQ6	t/s
93	DQ22	t/s
94	DQ5	t/s
95	DQ4	t/s
96	DQ3	t/s

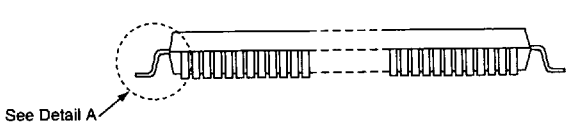
Table 1. S5933 Numerical Pin Assignment - 160 PQFP (Continued)

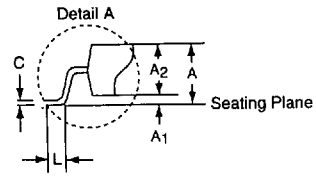
Pin#	Signal	Type
97	EA6	t/s
98	DQ2	t/s
99	DQ1	t/s
100	DQ0	t/s
101	EA7	t/s
102	WRFIFO#	in
103	WRFULL	out
104	RDFIFO#	in
105	DQ21	t/s
106	RDEMPTY	out
107	PTADR#	in
108	PTWR	out
109	EA8	t/s
110	VSS	V
111	VCC	V
112	PTBURST#	out
113	EA9	out
114	PTATN#	out
115	PTRDY#	in
116	PTBE0#	out
117	DQ20	t/s
118	PTBE1#	out
119	PTBE2#	out
120	PTBE3#	out
121	EA10	out
122	PTNUM1	out
123	PTNUM0	out
124	IRQ#	out
125	DQ19	t/s
126	SYSRST#	out
127	EWR#/SDA	t/s
128	ERD#/SCL	out

Pin#	Signal	Type
129	EA11	out
130	VSS	V
131	VCC	V
132	ADR6	in
133	DQ18	t/s
134	NC	—
135	SNV	in
136	NC	—
137	EA12	out
138	FLT#	in
139	RST#	in
140	BPCLK	out
141	EA13	out
142	CLK	in
143	GNT	in
144	REQ#	out
145	DQ17	t/s
146	AD31	t/s
147	AD30	t/s
148	AD29	t/s
149	EA14/FWE	t/s
150	VSS	V
151	VCC	V
152	AD28	t/s
153	EA15/FRF	t/s
154	AD27	t/s
155	AD26	t/s
156	AD25	t/s
157	DQ16	t/s
158	AD24	t/s
159	C/BE3#	t/s
160	IDSEL	in

Package Physical Dimensions - 160 PQFP







PQFP IN MILLIMETERS		
LEAD#	160	
SYMBOL	MIN	MAX
A	---	4.07
A1	0.25	---
A2	3.17	3.87
B	0.22	0.38
C	0.15	0.20
D1	27.90	28.10
E1	27.90	28.10
e	0.65	BSC
D	31.65	32.15
E	31.65	32.15
L	0.65	0.95

Table 2. S5933 Numerical Pin Assignment - 208 TQFP

Pin#	Signal	Type
1	VDD	V
2	VSS	V
3	VSS	V
4	EQ0	t/s
5	AD23	t/s
6	AD22	t/s
7	AD21	t/s
8	DQ31	t/s
9	AD20	t/s
10	AD19	t/s
11	AD18	t/s
12	N/C	---
13	EQ1	t/s
14	VSS	V
15	VSS	V
16	VDD	V
17	VDD	V
18	AD17	t/s
19	DQ30	t/s
20	AD16	t/s
21	C/BE2#	t/s
22	FRAME#	t/s
23	EQ2	t/s
24	IRDY#	t/s
25	TRDY#	t/s
26	DEVSEL#	t/s
27	EQ3	t/s
28	STOP#	t/s
29	LOCK#	I
30	PERR#	t/s
31	DQ29	t/s
32	SERR#	O

Pin#	Signal	Type
33	N/C	---
34	PAR	t/s
35	C/BE1#	t/s
36	EQ4	t/s
37	VSS	V
38	VSS	V
39	VDD	V
40	VDD	V
41	AD15	t/s
42	EQ5	t/s
43	AD14	t/s
44	AD13	t/s
45	AD12	t/s
46	DQ28	t/s
47	AD11	t/s
48	AD10	t/s
49	AD9	t/s
50	VDD	V
51	VDD	V
52	VDD	V
53	VDD	V
54	VSS	V
55	VSS	V
56	EQ6	t/s
57	AD8	t/s
58	C/BE0#	t/s
59	AD7	t/s
60	DQ27	t/s
61	AD6	t/s
62	AD5	t/s
63	AD4	t/s
64	N/C	---

Pin#	Signal	Type
65	EQ7	t/s
66	VSS	V
67	VSS	V
68	VDD	V
69	N/C	---
70	AD3	t/s
71	DQ26	t/s
72	AD2	t/s
73	AD1	t/s
74	AD0	t/s
75	EA0	t/s
76	INTA#	O
77	MODE	I
78	BE3	I
79	EA1	t/s
80	BE2	I
81	BE1	I
82	ADR5	I
83	DQ25	t/s
84	ADR4	I
85	N/C	---
86	ADR3	I
87	ADR2	I
88	EA2	t/s
89	VSS	V
90	VSS	V
91	VSS	V
92	VDD	V
93	RD#	I
94	EA3	t/s
95	WR#	I
96	SELECT#	I

Table 2. S5933 Numerical Pin Assignment - 208 TQFP (Continued)

Pin#	Signal	Type
97	DQ15	t/s
98	DQ24	t/s
99	VSS	V
100	DQ14	t/s
101	DQ13	t/s
102	DQ12	t/s
103	VDD	V
104	VDD	V
105	VDD	V
106	VSS	V
107	VSS	V
108	EA4	t/s
109	DQ11	t/s
110	DQ10	t/s
111	DQ9	t/s
112	DQ23	t/s
113	DQ8	t/s
114	BE0	I
115	DQ7	t/s
116	N/C	---
117	EA5	t/s
118	VSS	V
119	VSS	V
120	VDD	V
121	N/C	---
122	DQ6	t/s
123	DQ22	t/s
124	DQ5	t/s
125	DQ4	t/s
126	DQ3	t/s
127	EA6	t/s
128	DQ2	t/s

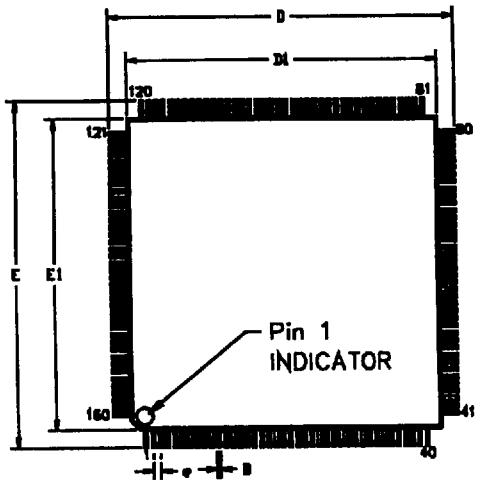
Pin#	Signal	Type
129	DQ1	t/s
130	DQ0	t/s
131	EA7	t/s
132	WRFIFO#	I
133	WRFULL	O
134	RDFIFO#	I
135	DQ21	t/s
136	RDEMPTY	O
137	N/C	---
138	PTADDR#	I
139	PTWR	O
140	EA8	t/s
141	VSS	V
142	VSS	V
143	VSS	V
144	N/C	---
145	PTBURST#	O
146	EA9	t/s
147	PTATN#	O
148	PTRDY#	I
149	PTBE0#	O
150	DQ20	t/s
151	PTBE1#	O
152	PTBE2#	O
153	PTBE3#	O
154	VDD	V
155	VDD	V
156	VDD	V
157	VDD	V
158	VSS	V
159	VSS	V
160	EA10	O

Pin#	Signal	Type
161	PTNUM1	O
162	PTNUM0	O
163	IRQ#	O
164	DQ19	t/s
165	SYSRST#	O
166	SDA/EWR	O
167	SCL/ERD	O
168	N/C	---
169	EA11	O
170	VSS	V
171	VSS	V
172	VDD	V
173	N/C	---
174	ADR6	I
175	DQ18	t/s
176	NV	I
177	N/C	---
178	EA12	O
179	FLT#	I
180	RST#	I
181	BPCLK	t/s
182	EA13	O
183	CLK	I
184	GNT#	I
185	REQ#	O
186	DQ17	t/s
187	AD31	t/s
188	AD30	t/s
189	N/C	---
190	AD29	t/s
191	EA14	O
192	VSS	V

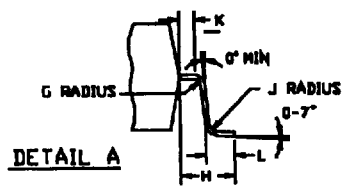
Table 2. S5933 Numerical Pin Assignment - 208 TQFP (Continued)

Pin#	Signal	Type
193	VSS	V
194	VDD	V
195	N/C	---
196	AD28	t/s
197	EA15	O
198	AD27	t/s
199	AD26	t/s
200	AD25	t/s
201	N/C	---
202	DQ16	t/s
203	AD24	t/s
204	C/BE3#	t/s
205	IDSEL	I
206	VDD	V
207	VDD	V
208	VDD	V

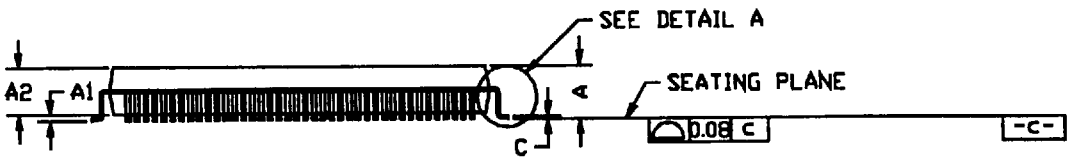
Package Physical Dimensions - 208 TQFP



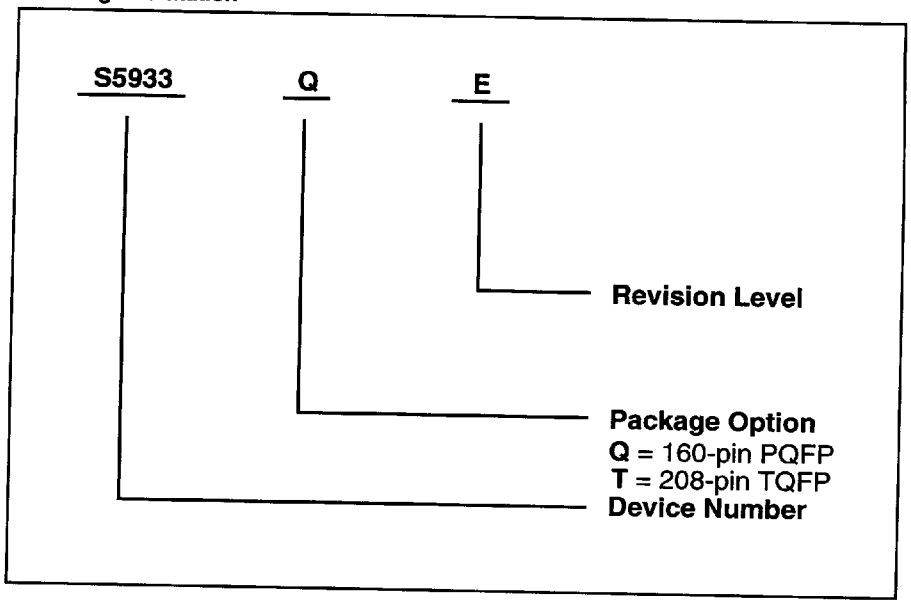
Symbol	MIN	NOM	MAX
A	-	-	1.60
A1	0.05	-	0.15
A2	1.35	1.40	1.45
D	30.00 BSC		
D1	28.00 BSC		
E	30.00 BSC		
E1	28.00 BSC		
L	0.45	0.60	0.75
e	0.50 BSC		
B	0.17	0.22	0.27
c	0.09	-	0.20
G	0.08	-	-
H	1.00 BSC		
J	0.08	-	0.20
K	0.20	-	-
2 H	2.00 BSC		



Ref. JEDEC - MO-136
ANAM AMKOR - DWG#34514



Ordering Information



INDEX

A

Access Time, Serial BIOS ROM 3-89

Accesses
- Asynchronous 3-105, 3-106
- nv RAM 3-106
- Synchronous 3-105, 3-106

Add-On
- Address (ADR[6:2]) 3-20
- Bus Interface 3-105
- Bus Size (MODE) 3-20, 3-105
- Byte Enables (BE[3:0]#) 3-20, 3-105
- FIFO Port 3-68
- General Control/Status Register 3-77
- Incoming Mailboxes 3-68
- Interrupt (IRQ#) 3-22, 3-105
- Interrupt Control/Status Register 3-74
- Master Transfer Count Enable 3-77
- Operation Registers 3-67
- Outgoing Mailboxes 3-68
- Pass-Thru Address Register 3-70
- Pass-Thru Data Register 3-70
- Read Strobe (RD#) 3-20, 3-105
- Reset (SYSRST#) 3-20, 3-105
- Write Strobe (WR#) 3-20, 3-105

B

Base Address Region, Assigning 3-40
Buffered PCI Clock (BPCLK) 3-22
Built-in Self Test (BIST) 3-39
Burst Order 3-92
Burst Transfers, PCI Bus 3-92
Bus Master Control/Status 3-63
Bus Master Enable 3-27
Bus Master Read Address 3-55
Bus Master Read Transfer Count 3-56
Bus Master Write Address 3-53
Bus Master Write Transfer Count 3-54

Bus Mastering
- Add-On Initiated 3-109
- AMREN Control 3-109
- AMWEN Control 3-109
- PCI Bus 3-101

C

Cache Line Size 3-36
Class Codes 3-32, 3-33, 3-34, 3-35
Command Register 3-27

D

Device ID 3-26

E

Expansion BIOS 3-88, 3-111
Expansion ROM Base Address 3-44, 3-111

F

Fast Back-to-Back Cycles 3-26

FIFO
- 8/16-Bit Add-On Interface 3-134
- 16-Bit Endian Conversion 3-124
- Add-On Accesses Asynchronous 3-131
- Add-On Accesses Synchronous 3-132
- Bus Mastering 3-126
- Byte Lane Steering 3-134
- Configuration at Reset 3-135
- Control Signals 3-21, 3-126, 3-133
- Direct Read (RDFIFO#) 3-21, 3-109, 3-132
- Direct Write (WRFIFO#) 3-21, 3-109, 3-132
- DMA Transfer Address 3-127
- DMA Transfer Byte Count 3-127
- Error Condition Interrupts 3-128
- Flag Reset (Inputs) 3-109
- Flag Reset (Software) 3-63, 3-77
- Management 3-123, 3-127
- Overview 3-123
- PCI Reads 3-131
- PCI Writes 3-131
- Status Indicators (Outputs) 3-21, 3-109, 3-121, 3-133
- Status Indicators (Software) 3-63, 3-77, 3-135

FRAME#/IRDY# Valid Combinations 3-102

H

Header Type 3-38

I

I/O Access Enable 3-27
Initialization, S5933 3-83
Initiator Ready (IRDY#) 3-17

Interrupt
- BIST 3-39
- Bus Master Error 3-74
- Enabling 3-59, 3-74
- Hardware, to PCI 3-109
- Mailbox 3-59, 3-74
- Master Abort 3-59
- PCI Bus 3-104
- Read Transfer Count 3-59, 3-74
- Target Abort 3-59
- Write Transfer Count 3-59, 3-74

Interrupt Control/Status Register 3-59
Interrupt Handler, PC 4-53
Interrupt Line Register 3-46
Interrupt Pin Register 3-47

L

Latency Components, Bus Mastering
- Bus Acquisition 3-102
- Bus Arbitration 3-101
- Target Latency 3-102

Latency Timer 3-37, 3-96
Locking a Target 3-102, 3-103

M

Mailbox
- 8/16-Bit Add-On Interface 3-117
- Block Diagrams 3-115
- Empty/Full Status 3-57, 3-72, 3-116
- Enabling Add-On Interrupts 3-120
- Enabling PCI Interrupts 3-119
- Flag Reset 3-63, 3-77, 3-116
- Incoming 3-115
- Interlocking Mechanism 3-115
- Interrupts 3-109, 3-116, 3-119
- Mailbox 4, Byte 3 3-109, 3-116
- Monitoring Status 3-118
- PCI Interrupt (Direct) 3-116
- PCI Operation Registers 3-117
- Reading Add-On Incoming 3-119
- Reading PCI Incoming 3-118
- Status Polling 3-118
- Writing Add-On Outgoing 3-119
- Writing PCI Outgoing 3-118

Master Abort 3-29, 3-97
Master-Initiated Termination 3-95
Maximum Latency Register 3-49
Memory Access Enable 3-27
Memory Write/Invalidate 3-28, 3-36
Minimum Grant Register 3-48

N

Normal PCI Cycle Completion 3-95
nv RAM
- Accessing 3-63, 3-77, 3-110
- Block Diagram 3-9
- Interface Timing 3-111, 3-113
- Loading from Byte-Wide 3-83
- Loading from Serial 3-84
- Overview 3-9
- Read Operation 3-113
- Read Strobe (ERD#) 3-19
- Valid Boot Image 3-84
- Write Operation 3-114
- Write Strobe (EWR#) 3-19

O

Operation Register Access Timing
- Asynchronous RDFIFO# 3-165
- Asynchronous WRFIFO# 3-166
- Synchronous RDFIFO# 3-167
- Synchronous WRFIFO# 3-168
- Asynchronous RD# 3-169
- Asynchronous WR# 3-170
- Synchronous RD# 3-171
- Synchronous WR# 3-173
- Interrupt, Add-On 3-179
- Mailbox 4, Byte 3 3-179
- nv RAM Read 3-178
- nv RAM Write 3-178
- Pass-Thru Read 3-176
- Pass-Thru Status 3-177
- Pass-Thru Write 3-176

P

Parity Error Enable 3-27
Parity Error Signals, PCI 3-37
Pass-Thru
- 8/16-Bit Add-On Interface 3-154
- Accessing a Region 3-158
- Add-On Burst Read 3-149
- Add-On Burst Write 3-145
- Add-On Bus Interface 3-142
- Add-On Bus Width 3-42, 3-140, 3-157
- Add-On Disconnect Operation 3-153
- Address Register 3-70, 3-109, 3-139
- Base Address Registers 3-154, 3-157
- Block Diagram 3-12
- Bursting Beyond Region 3-141
- Byte Lane Steering 3-154
- Control Inputs 3-21, 3-108, 3-140
- Creating a Region 3-157
- Data Bus Steering 3-140
- Data Register 3-70, 3-109, 3-129
- I/O Mapped Region 3-157
- Memory Mapped Region 3-157
- Overview 3-139
- PCI Burst Access 3-141
- PCI Bus Configuration 3-157
- PCI Read Retries 3-142
- PCI Single Cycle Access 3-140
- PCI Write Retries 3-141
- Physical Address 3-158
- Region Definition 3-157
- Retries by Initiator 3-156
- Status Indicators 3-21, 3-108, 3-140
- Target Retries 3-153, 3-154
- Transfers 3-139

PCI
- Agent 3-12
- Bus Commands 3-16, 3-91
- Bus Parity (PAR) 3-16
- Bus Request (REQ#) 3-18
- Bus Transactions 3-91
- Clock (CLK) 3-17
- Configuration Cycles 3-86
- Configuration Registers 3-23
- Incoming Mailboxes 3-52

Interrupt (INTA#) 3-18, 3-47
Local Bus 3-9
Operation Registers 3-51
Outgoing Mailboxes 3-52
Parity Error (PERR#) 3-18
Reset (RST#) 3-17
Status Register 3-29, 3-97
System Error (SERR#) 3-18

R

Read Transfers, PCI Bus 3-92, 3-93
Revision ID 3-31

S

S5933 Add-On Signals
ADR[6:2] 3-20
ADR1 3-20
AMREN 3-109
AMWEN 3-109
BE[2:0]# 3-20
BPCLK 3-22
DQ[31:00] 3-20
EMBCLK 3-117
FLT# 3-22
FRC# 3-109
FRF 3-126
FWC# 3-109
FWE 3-126
IRQ# 3-22
MODE 3-20
PTADR# 3-21
PTATN# 3-21
PTBE[3:0]# 3-21
PTBURST# 3-21
PTNUM[1:0] 3-21
PTRDY# 3-21
PTWR 3-21
RD# 3-20
RDEMPTY 3-21
RDFIFO# 3-21
SELECT# 3-20
SYSRST# 3-22
WR# 3-20
WRFIFO# 3-21
WRFULL 3-21

S5933 nv RAM Signals
EA[15:0] 3-19
EQ[7:0] 3-19
ERD# 3-19
EWR# 3-19
SCL 3-19
SDA 3-19
SNV 3-19

S5933 PCI Signals
AD[31:00] 3-16
C/BE[3:0]# 3-16
CLK 3-17
DEVSEL# 3-17
FRAME# 3-17
GNT# 3-18
IDSEL 3-17
INTA# 3-18
IRDY# 3-17
LOCK# 3-17
PAR 3-16
PERR# 3-18
REQ# 3-18
RST# 3-17
SERR# 3-18
STOP# 3-17
TRDY# 3-17

S5933 PCI Cycle Support 3-91
S5933 Pin Diagram 3-15
Serial Clock (SCL) 3-19
Serial Data (SDA) 3-19
System Error Status 4-28

T

Target Abort 3-28, 3-99
Target Disconnect 3-98
Target-Initiated Termination 3-97, 3-99, 3-99
Target Ready (TRDY#) 3-17
Target Requested Retries 3-99
Targets, Slow Responding 3-98
Transfer Count Status 3-63, 3-77

V

Vendor ID 3-25

W

Write Transfers, PCI Bus 3-94