21555 Non-Transparent PCI-to-PCI Bridge

User Manual

July 2001

Order Number: 278321–002

Contents

Figures

Tables

Preface 1

A brief description of the contents of this manual follows.

Chapter 1, “Preface”	Provides information about the contents and organization of this book.
Chapter 2, “Introduction”	Provides an overview of the 21555 functionality and architecture.
Chapter 3, “Signal Descriptions”	Describes PCI signal pins grouped by function.
Chapter 4, “Address Decoding”	Contains details about how addresses are decoded.
Chapter 5, “PCI Bus Transactions”	Describes how the 21555 implements the theory of operation about PCI transactions.
Chapter 6, “Initialization Requirements”	Describes the reset operation and initialization requirements.
Chapter 7, “Clocking”	Describes 21555 clocking support.
Chapter 8, “Parallel ROM Interface”	Describes the 21555 Parallel ROM Interface.
Chapter 9, “Serial ROM Interface”	Describes the 21555 Serial ROM Interface.
Chapter 10, “Arbitration”	Explains how 21555 implements primary and secondary PCI bus arbitration.
Chapter 11, “Interrupt and Scratchpad Registers”	Describes interrupt support and scratchpad registers.
Chapter 12, “Error Handling”	Describes parity error responses and system error reporting.
Chapter 13, “JTAG Test Port”	Explains the implementation of JTAG test port.
Chapter 14, “I2O Support”	Explains how the 21555 implements an I20 messaging unit.
Chapter 15, “VPD Support”	Describes Vital Product Data support through SROM interface.
Chapter 16, “List of Registers”	This chapter contains all of the 21555 register information and contains a register summary.
Appendix A, “Acronyms”	Definition of terms used in this book.

1.1 Cautions and Notes

Caution: Cautions provide information to prevent damage to equipment or loss of data.

Note: Notes emphasize particularly important information.

1.2 Data Units

This manual uses the following data-unit terminology.

Term	Words	Bytes	Bits
Byte	½	1	8
Word	1	2	16
Dword	2	4	32
Quadword	4	8	64

1.3 Numbering

All numbers are decimal unless appended with a radix specifier.

- “h” is the hexadecimal radix.
- “b” is the binary radix.

A range of (numbers, bits, bytes, addresses and so on) within a larger group of numbers is specified with colon (:). The range may be enclosed in [square brackets] as well.

- The left number is upper limit of the range.
- The right number is the lower limit of the range.
  For example: Primary byte offset: 13:10h.

1.4 Signal Nomenclature

21555 device signal names are printed in lowercase type. Prefixes and suffixes are tagged with a leading or trailing letter and are delimited with an "_" underscore:

- The prefix "p_" denotes a primary bus signal. For example: **p_ad** is the primary interface address/data bus.
- The prefix "s_" denotes a secondary bus signal. For example: **s_ad** is the secondary interface address/data bus.
- Other prefixes might appear. l_(load), pr_(parallel rom), and so on.
- The suffix "_l" means that the condition is qualified when that signal is low or approximately zero (0) volts. For example: **p_frame_l** is a low-asserted signal.
- If no suffix is applied, it means that the condition is qualified when the signal is high or approximately equal to **vcc**. For example: **p_idsel** is a high-asserted signal.

PCI signals that can be on either the primary interface or the secondary interface are printed in uppercase, normal type. The names of low-asserted signals are followed by #. For example, "asserting **FRAME#**" can refer to the assertion of the **p_frame_l** signal when the transaction is occurring on the primary bus or the assertion of the **s_frame_l** signal when the transaction is occurring on the secondary bus.

Table 1 describes the Signal Type letters that appear in Table 6 through Table 10 in Chapter 3, "Signal Descriptions".

Table 1. Signal Type Abbreviations

Signal Type	Description
I	Standard input only.
O	Standard output only.
TS	Tristate bidirectional.
STS	Sustained tristate. Active low signal must be pulled high for one clock cycle when deasserting.
OD	Standard open drain.

1.5 Register Abbreviations

When a register is associated with the primary interface, its name is preceded with *Primary*. When a register is associated with the secondary interface, its name is preceded with *Secondary*. When a register is shared by both interfaces, it is not preceded with *Primary* or *Secondary*. The byte offsets at which each register can be accessed from each interface are listed in each register description.

Table 2 lists the register access abbreviations.

Table 2. Register Abbreviations

Access Type	Description
DCA	Downstream configuration address.
DCD	Downstream configuration data.
DIA	Downstream I/O address.
DID	Downstream I/O data
R	Read only. Writes have no effect.
R/W	Read/Write.
R/W1TC	Read. Write 1 to clear.
R/W1TS	Read. Write 1 to set.
R0TS	Read 0 to set.
R/(WS)	Read. Write from secondary interface only. Primary bus writes have no effect.
R/(WP)	Read. Write from primary interface only. Secondary bus writes have no effect.
UCA	Upstream configuration address.
UCD	Upstream configuration data.
UIA	Upstream I/O address.
UID	Upstream I/O data.
W1TL	Write 1 to load.

Introduction 2

The Intel® 21555 is a PCI peripheral device that performs PCI bridging functions for embedded and intelligent I/O applications. The 21555 has a 64-bit primary interface, a 64-bit secondary interface, and 66-MHz capability. In this document the 21555 non-transparent device is compared to the related 21154 transparent devices. Both devices have similar operating characteristic.

The 21555 is a “non-transparent” PCI-to-PCI Bridge (PPB) that acts as a gateway to an intelligent subsystem. It allows a local processor to independently configure and control the local subsystem. The 21555 implements an I20 message unit that enables any local processor to function as an intelligent I/O processor (IOP) in an I20-capable system. Since the 21555 is architecture independent, it works with any host and local processors that support a PCI bus. This architecture independence enables vendors to leverage existing investments while moving products to PCI technology.

Unlike a transparent PPB, the 21555 is specifically designed to bridge between two processor domains. The processor domain on the primary interface of the 21555 is referred to as the host domain, and its processor is the host processor. The secondary bus interfaces to the local domain and the local processor. Special features include support of:

- Independent primary and secondary PCI clocks. See Chapter 7.
- Independent primary and secondary address spaces.
- Address translation between the primary (host) and secondary (local) domains. See Chapter 4.

The 21555 allows add-in card vendors to present a higher level of abstraction to the host system than is possible with a transparent PPB. The 21555 uses a Type 0 configuration header, which presents the entire subsystem as a single “device” to the host processor. This allows loading of a single device driver for the entire subsystem, and independent local processor initialization and control of the subsystem devices. Since the 21555 uses a Type 0 configuration header, it does not require hierarchical PPB configuration code.

The 21555 forwards transactions between the primary and secondary PCI buses as does a transparent PPB. In contrast to a transparent PPB, the 21555 can translate the address of a forwarded transaction from a system address to a local address, or vice versa. This mechanism allows the 21555 to hide subsystem resources from the host processor and to resolve any resource conflicts that may exist between the host and local subsystems.

The 21555 operates at 3.3 V, but is also 5.0-V I/O tolerant. Adapter cards designed for the 21555 can be keyed as a PCI universal card edge connector, permitting use in either a 5-V or 3-V slot.

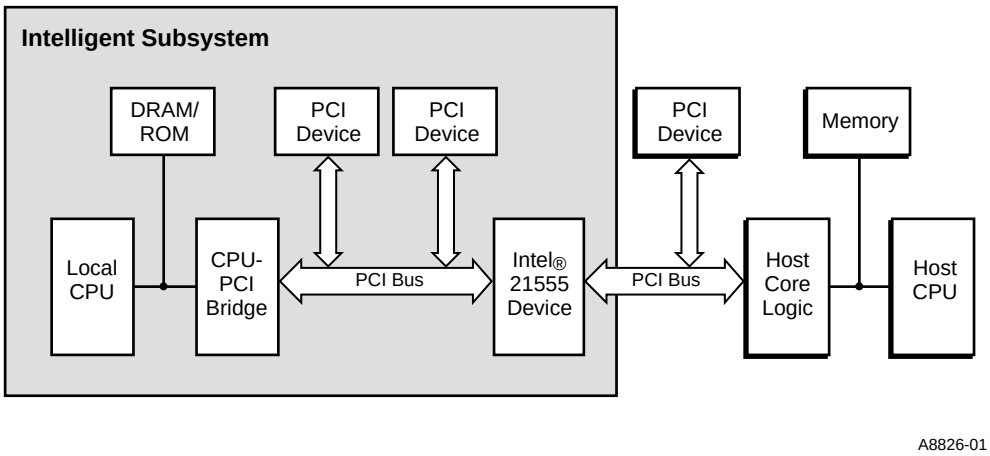
2.1 Comparing a 21555 to a Transparent PPB

The 21555 is functionally similar to a transparent PPB in that both provide a connection path between devices attached to two independent PCI buses. A 21555 and a PPB allow the electrical loading of devices on one PCI bus to be isolated from the other bus while permitting concurrent operation on both buses. Since the *PCI Local Bus Specification* restricts PCI option cards to a single electrical load, the ability of PPBs and the 21555 to spawn PCI buses enables the design of multi-device PCI option cards. The key difference between a PPB and the 21555 is that the presence of a PPB in a connection path between the host processor and a device is transparent to devices and device drivers, while the presence of the 21555 is not. This difference enables the 21555 to provide features that better support the use of intelligent controllers in the subsystem.

A primary goal of the PPB architecture is that PPB are transparent to devices and device drivers. For example, no changes are needed to a device driver when a PCI peripheral is located behind a PPB. Once configured during system initialization, a PPB operates without the aid of a device driver. A PPB does not require a device driver of its own since it does not have any resources that must be managed by software during run-time. This requirement for transparency forced the usage of a flat addressing model across PPBs. This means that a given physical address exists at only one location in the PCI bus hierarchy and that this location may be accessed by any device attached at any point in the PCI bus hierarchy. As a consequence, it is not possible for a PPB to isolate devices or address ranges from access by devices on the opposite interface of a PPB. The PPB architecture assumes that the resources of any device in a PCI system are configured and managed by the host processor.

Figure 1 shows a hypothetical PCI add-in card used for an intelligent controller application. In some applications the transparency of a PPB is not desired. For example,

Figure 1. 21555 Intelligent Controller Application



Assume:

- That the local processor on the add-in card is used to manage the resources of the devices attached to the add-in card's local PCI bus.
- That it is desirable to restrict access to these same resources from other PCI bus masters in the system and from the host processor.
- That there is a need to resolve address conflicts that may exist between the host system and the local processor.

The non transparency of the 21555 is perfectly suited to this kind of configuration, where a transparent PPB would be problematic.

Since the 21555 is non transparent, the device driver for the add-in card must be aware of the presence of the 21555 and manage its resources appropriately. The 21555 allows the entire subsystem to appear as a single virtual device to the host. This enables configuration software to identify the appropriate driver for the subsystem.

With a transparent PPB, a driver does not need to know about the presence of the bridge and manage its resources. The subsystem appears to the host system as individual PCI devices on a secondary PCI bus, not as a single virtual device.

Table 3 shows compares a 21555 and to a transparent PPB.

Table 3. 21555 and PPB Feature Comparison

Feature	Non-Transparent PPB or 21555	Transparent PPB
Transaction forwarding	• Adheres to PPB ordering rules. • Uses posted writes and delayed transactions. • Adheres to PPB transaction error and parity error guidelines, although some errors may be reported differently.	• Adheres to PPB ordering rules. • Uses posted writes and delayed transactions. • Adheres to PPB transaction error and parity error guidelines.
Address decoding	• Base address registers (BARs) are used to define independent downstream and upstream forwarding windows. • Inverse decoding is only used for upstream transactions above the 4GB boundary.	• PPB base and limit address registers are used to define downstream forwarding windows. • Inverse decoding for upstream forwarding.
Address translation	Supported for both memory and I/O transactions.	None. Flat address model is assumed.
Configuration	• Downstream devices are not visible to host. • Does not require hierarchical configuration code (Type 0 configuration header). • Does not respond to Type 1 configuration transactions. • Supports configuration access from the secondary bus. • Implements separate set of configuration registers for the secondary interface.	• Downstream devices are visible to host. • Requires hierarchical configuration code (Type 1 configuration header). • Forwards and converts Type 1 configuration transactions. • Does not support configuration access from the secondary bus. Same set of configuration registers is used to control both primary and secondary interfaces.
Run-time resources	Includes features such as doorbell interrupts, I20 message unit, and so on, that must be managed by the device driver.	Typically has only configuration registers; no device driver is required.
Clocks	Generates secondary bus clock output. Asynchronous secondary clock input is also supported.	Generates one or more secondary bus clock outputs.
Secondary bus central functions	Implements secondary bus arbiter. This function can be disabled. Drives secondary bus **AD**, **C/BE#**, and **PAR** during reset. This function can be disabled.	Implements secondary bus arbiter. Drives secondary bus **AD**, **C/BE#**, and **PAR** during reset.

2.2 Architectural Overview

This section describes the buffers, registers, and control logic of the 21555:

2.2.1 Data Buffers

Data buffers include the buffers along with the associated data path control logic. Delayed transaction buffers contain the compare functionality for completing delayed transactions. The blocks also contain the watchdog timers associated with the buffers. The data buffers are as follows:

- Four-entry downstream delayed transaction buffer.
- Four-entry upstream delayed transaction buffer.
- 256-byte downstream posted write buffer.
- 256-byte upstream posted write buffer.
- 256-byte downstream read data buffer.
- 256-byte upstream read data buffer.
- Two downstream I20 delayed transaction entries.

2.2.2 Registers

The following register blocks also contain address decode and translation logic, I20 message unit, and interrupt control logic:

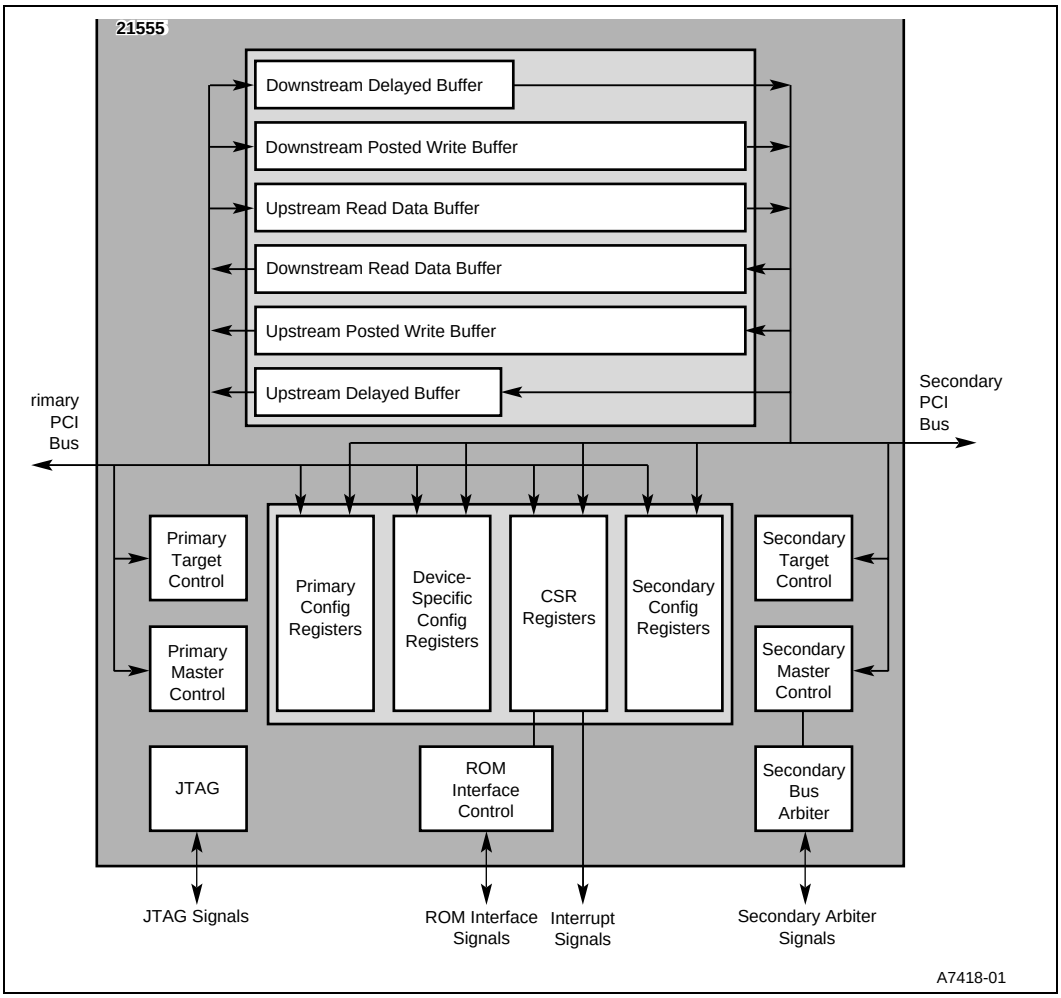
- Primary interface header Type 0 configuration registers.
- Secondary interface header Type 0 configuration registers.
- Device-specific configuration registers.
- Memory and I/O mapped control and status registers.

2.2.3 Control Logic

- The 21555 has the following control logic:
- Primary PCI target control logic.
- Primary PCI master control logic.
- Secondary PCI target control logic.
- Secondary PCI master control logic.
- ROM interface control logic for both serial and parallel ROM connections (interfaces between the ROM registers and ROM signals).
- Secondary PCI bus arbiter interface to secondary bus device request and grant lines, as well as the 21555 secondary master control logic.
- JTAG control logic.

Figure 2 shows the 21555 microarchitecture.

Figure 2. 21555 Microarchitecture



2.3 Special Applications

2.3.1 Primary Bus VGA Support

The 21555 provides hardware support that allows configuration of itself as a Video Graphics Adapter (VGA) device. The primary class code should be preloaded through the serial ROM (SROM) or loaded by the local processor with the value for a VGA device (Base Class 03h, Sub-Class 00h, Programming Interface 00h). This allows the 21555 to present itself to the host as a VGA device.

The VGA Mode field in the Chip 0 Control register (see page 156) should be set to 01b to enable decoding of VGA transactions on the primary bus for forwarding to the secondary bus. These bits can be set through SROM preload, or either from a primary or secondary bus configuration write. Table 4 gives addresses that are decoded.

Table 4. Decoded and Not Decoded Addresses

Memory addresses	[000BFFFFh : 000A0000h]
I/O addresses:	AD[9:0]3BBh:3B0h 3DFh:3C0h
Bits not decoded.	AD[31:16]000h (No address translation is performed on these addresses.)

The 21555 cannot be enabled as a snooping agent on the primary bus. This is because the 21555 cannot guarantee that it can buffer and forward all palette writes, since the 21555 has finite buffer space and no backoff mechanism when snooping. The 21555 should not be configured to appear as a VGA device in those applications where it may try to configure the 21555 as a snooping agent.

The parallel ROM can be used to store VGA BIOS code, which is mapped through the Primary Expansion ROM BAR.

2.3.2 Secondary Bus VGA Support

The 21555 can be enabled to decode VGA transactions on the secondary bus for forwarding to the primary bus. This is done by setting the VGA Enable field in the Chip Control 0 register to 10b. The addresses that are decoded are the same as for the primary VGA decode, and again the addresses are not translated.

Upstream forwarding of VGA transactions can be useful for applications that want to allow access to a primary bus VGA device frame buffer by local processors in intelligent I/O or embedded subsystems.

Note: VGA decoding must *not* be enabled for both the primary and secondary interface. The value 11b is illegal for the VGA Enable field and can yield unpredictable results.

2.4 Programming Notes

2.4.1 Addressing

The non-transparent addressing model that the 21555 uses can cause problems if not programmed properly. Programming errors include:

- Setting a translated base address for a downstream range to fall within an address range defined for upstream forwarding. This would cause the 21555 to respond as a target on the secondary bus to a downstream transaction that it has initiated as a master. The transaction would then be forwarded back to the primary bus. The address on the primary bus depends on the translated base address value for that upstream range.
- Setting a translated base address for an upstream range to fall within an address range for downstream forwarding. This results in similar behavior described in the previous condition, but in the opposite direction.
- Enabling I/O subtractive decoding in both directions. When an I/O transaction is subtractively decoded on the primary bus and forwarded downstream by the 21555, and no target responds on the secondary bus, the 21555 subtractively decodes the transaction on the secondary bus and forwards it back upstream. Since there is no address translation for subtractively decoded I/O transactions, this results in the 21555 forwarding the transaction downstream and upstream forever.
- Enabling VGA decoding in both directions. Refer to subtractive I/O decoding in the previous bullet. Again, there is the case of a non translated I/O address decoded by the 21555 on both interfaces as a target and forwarded to the opposite interface.

2.4.2 Transaction Forwarding

When using the indirect I/O transaction generation mechanism, the low two bits of the I/O address in the I/O Address register must match the byte enables as described in the *PCI Local Bus Specification, Revision 2.2*. The 21555 does not correct any discrepancies between the byte enables and address bits [1:0].

2.4.3 ROM Access

Parallel and SROM access mechanisms do not accommodate multiple masters. That is, when more than one master attempts to access the ROM during the same time period, wrong data may be returned or written to the ROM. There is no semaphore method to guarantee atomicity of the ROM address, data, and control register accesses.

This also applies to a parallel ROM access through the Primary Expansion ROM BAR at the same time a secondary bus master might be accessing ROM registers.

Signal Descriptions 3

This chapter presents the theory of operation information about the PCI signal interface. See Chapter 16 for specific information about PCI registers. Table 5 describes the PCI signal groups, function, and provides a page reference.

Table 5. Signal Pin Functional Groups

Group by Signal Pin	Description	See Page
Primary Bus and Extension interface Signal Pins	All PCI pins required by the *PCI Local Bus Specification, Revision 2.2.*	Section 3.1 on page 24
	All PCI 64-bit extension pins required by the *PCI Local Bus Specification, Revision 2.2.*	Section 3.2 on page 26
Secondary Bus and Extension Interface Signal Pins	All PCI pins required by the *PCI Local Bus Specification, Revision 2.2.*	Section 3.3 on page 28
	All PCI 64-bit extension pins required by the *PCI Local Bus Specification, Revision 2.2.*	Section 3.4 on page 30
Miscellaneous Signal Pins	Two input voltage signaling pins.	Section 3.5 on page 31
	Power Management, Hot-Swap, and Reset Signals	Section 6.1 on page 65
Timing	Primary and Secondary PCI Bus Clock Signals	Section 7.1 on page 77
Optional configuration and expansion memory.	Interface Signals	Section 8.1 on page 81
	SROM Interface Signals	Section 9.1 on page 91
Arbitration	Primary PCI Bus Arbitration Signals	Section 10.1 on page 97
Interrupt	Primary and Secondary PCI Bus Interrupt Signals	Section 11.1 on page 101
Error	Error Signals	Section 12.1 on page 105
Test Access Port	JTAG Signals	Section 13.1 on page 111

3.1 Primary PCI Bus Interface Signals

Table 6 describes the primary PCI bus interface signals. The letters in the "Type" column are described in Table 1.

Table 6. Primary PCI Bus Interface Signals (Sheet 1 of 2)

Signal Name	Type	Description
p_ad[31:0]	TS	Primary PCI interface address and data. These signals are a 32-bit multiplexed address and data bus. During the address phase or phases of a transaction, the initiator drives a physical address on **p_ad[31:0]**. During the data phases of a transaction, the initiator drives write data, or the target drives read data, on **p_ad[31:0]**. When the primary PCI bus is idle, the 21555 drives **p_ad** to a valid logic level when **p_gnt_l** is asserted.
p_cbe_l[3:0]	TS	Primary PCI interface command and byte enables. These signals are a multiplexed command field and byte enable field. During the address phase or phases of a transaction, the initiator drives the transaction type on **p_cbe_l[3:0]**. When there are two address phases, the first address phase carries the dual-address command and the second address phase carries the transaction type. For both read and write transactions, the initiator drives byte enables on **p_cbe_l[3:0]** during the data phases. When the primary PCI bus is idle, the 21555 drives **p_cbe_l** to a valid logic level when **p_gnt_l** is asserted.
p_devsel_l	STS	Primary PCI interface **DEVSEL#**. Signal **p_devsel_l** is asserted by the target, indicating that the device is responding to the transaction. As a target, the 21555 decodes the address of a transaction initiated on the primary bus to determine whether to assert **p_devsel_l**. As an initiator of a transaction on the primary bus, the 21555 looks for the assertion of **p_devsel_l** within five clock cycles of **p_frame_l** assertion; otherwise, the 21555 terminates the transaction with a master abort. Upon completion of a transaction, **p_devsel_l** is driven to a deasserted state for one clock cycle and is then sustained by an external pull-up resistor.
p_frame_l	STS	Primary PCI interface **FRAME#**. Signal **p_frame_l** is driven by the initiator of a transaction to indicate the beginning and duration of an access on the primary PCI bus. Signal **p_frame_l** assertion (falling edge) indicates the beginning of a PCI transaction. While **p_frame_l** remains asserted, data transfers can continue. The deassertion of **p_frame_l** indicates the final data phase requested by the initiator. Upon completion of a transaction, **p_frame_l** is driven to a deasserted state for one clock cycle and is then sustained by an external pull-up resistor.
p_idsel	I	Primary PCI interface **IDSEL**. Signal **p_idsel** is used as the chip select line for Type 0 configuration accesses to 21555 configuration space from the primary bus. When **p_idsel** is asserted during the address phase of a Type 0 configuration transaction, the 21555 responds to the transaction by asserting **p_devsel_l**.
p_irdy_l	STS	Primary PCI interface **IRDY#**. Signal **p_irdy_l** is driven by the initiator of a transaction to indicate the initiator's ability to complete the current data phase on the primary PCI bus. During a write transaction, assertion of **p_irdy_l** indicates that valid write data is being driven on the **p_ad** bus. During a read transaction, assertion of **p_irdy_l** indicates that the initiator is able to accept read data for the current data phase. Once asserted during a given data phase, **p_irdy_l** is not deasserted until the data phase completes. Upon completion of a transaction, **p_irdy**_l is driven to a deasserted state for one clock cycle and is then sustained by an external pull-up resistor.

Table 6. Primary PCI Bus Interface Signals (Sheet 2 of 2)

Signal Name	Type	Description
p_par	TS	Primary PCI interface parity. Signal **p_par** carries the even parity of the 36 bits of **p_ad[31:0]** and **p_cbe_l[3:0]** for both address and data phases. Signal **p_par** is driven by the same agent that drives the address (for address parity) or the data (for data parity). Signal **p_par** contains valid parity one clock cycle after the address is valid (indicated by assertion of **p_frame_l**), or one clock cycle after the data is valid (indicated by assertion of **p_irdy_l** for write transactions and **p_trdy_l** for read transactions). Signal **p_par** is tristated one clock cycle after the **p_ad** lines are tristated. The device receiving data samples **p_par** as an input to check for possible parity errors. When the primary PCI bus is idle, the 21555 drives **p_par** to a valid logic level when **p_gnt_l** is asserted (one clock cycle after the **p_ad** bus is parked).
p_req_l	TS	Primary PCI bus **REQ#**. Signal **p_req_l** is asserted by the 21555 to indicate to the primary bus arbiter that it wants to start a transaction on the primary bus. Signal **p_req_l** is tristated during the assertion of chip reset.
p_stop_l	STS	Primary PCI interface **STOP#.** Signal **p_stop_l** is driven by the target of a transaction, indicating that the target is requesting the initiator to stop the transaction on the primary bus. • When **p_stop_l** is asserted in conjunction with **p_trdy_l** and **p_devsel_l** assertion, a disconnect with data transfer is being signaled. • When **p_stop_l** and **p_devsel_l** are asserted, but **p_trdy_l** is deasserted, a target disconnect without data transfer is being signaled. When this occurs on the first data phase, that is, no data is transferred during the transaction, this is referred to as a target retry. • When **p_stop_l** is asserted and **p_devsel_l** is deasserted, the target is signaling a target abort. Upon completion of a transaction, **p_stop_l** is driven to a deasserted state for one clock cycle and is then sustained by an external pull-up resistor.
p_trdy_l	STS	Primary PCI interface **TRDY#**. Signal **p_trdy_l** is driven by the target of a transaction to indicate the target's ability to complete the current data phase on the primary PCI bus. During a write transaction, assertion of **p_trdy_l** indicates that the target is able to accept write data for the current data phase. During a read transaction, assertion of **p_trdy_l** indicates that the target is driving valid read data on the **p_ad** bus. Once asserted during a given data phase, **p_trdy_l** is not deasserted until the data phase completes. Upon completion of a transaction, **p_trdy_l** is driven to a deasserted state for one clock cycle and is then sustained by an external pull-up resistor.

3.2 Primary PCI Bus Interface 64-Bit Extension Signals

Table 7 describes the primary PCI bus interface 64-bit extension signals. The letters in the “Type” column are described in Table 1.

Table 7. Primary PCI Bus Interface 64-Bit Extension Signals (Sheet 1 of 2)

Signal Name	Type	Description
p_ack64_l	STS	Primary PCI interface acknowledge 64-bit transfer. Signal **p_ack64_l** should never be driven when **p_req64_l** is not driven. Signal **p_ack64_l** is asserted by the target only when **p_req64_l** is asserted by the initiator, to indicate the target’s ability to transfer data using 64 bits. Signal **p_ack64_l** has the same timing as **p_devsel_l**. When deasserting, **p_ack64_l** is driven to a deasserted state for one clock cycle and is then sustained by an external pull-up resistor.
p_ad[63:32]	TS	Primary PCI interface address and data upper 32 bits. The 21555 does not bus park these pins. These pins are tristated during the assertion of **p_rst_l**. Signals **p_ad[63:32]** are driven to a valid value when the 64-bit extension is disabled (**p_req64_l** is deasserted during **p_rst_l** assertion). This multiplexed address and data bus provides an additional 32 bits to the primary interface. During the address phase or phases of a transaction, when the dual-address command is used and **p_req64_l** is asserted, the initiator drives the upper 32 bits of a 64-bit address; otherwise, these bits are undefined, and the initiator drives a valid logic level onto the pins. During the data phases of a transaction, the initiator drives the upper 32 bits of 64-bit write data, or the target drives the upper 32 bits of 64-bit read data, when **p_req64_l** and **p_ack64_l** are both asserted. When not driven, signals **p_ad[63:32]** are pulled up to a valid logic level through external resistors.
p_cbe_l[7:4]	TS	Primary PCI interface command and byte enables upper 4 bits. The 21555 does not bus park these pins. These pins are tristated during the assertion of **p_rst_l**. Signals **p_cbe_l[7:4]** are driven to a valid value when the 64-bit extension is disabled (**p_req64_l** is deasserted during **p_rst_l** assertion). These signals are a multiplexed command field and byte enable field. During the address phase or phases of a transaction, when the dual-address command is used and **p_req64_l** is asserted, the initiator drives the transaction type on **p_cbe_l[7:4]**; otherwise, these bits are undefined, and the initiator drives a valid logic level onto the pins. For both read and write transactions, the initiator drives byte enables for the **p_ad[63:32]** data bits on **p_cbe_l[7:4]** during the data phases, when **p_req64_l** and **p_ack64_l** are both asserted. When not driven, signals **p_cbe_l[7:4]** are pulled up to a valid logic level through external resistors.

Table 7. Primary PCI Bus Interface 64-Bit Extension Signals (Sheet 2 of 2)

Signal Name	Type	Description
p_par64	TS	Primary PCI interface upper 32 bits parity. The 21555 does not bus park this pin. This pin is tristated during the assertion of **p_rst_l**. Signal **p_par64** is driven to a valid value when the 64-bit extension is disabled (**p_req64_l** is deasserted during **p_rst_l** assertion). Signal **p_par64** carries the even parity of the 36 bits of **p_ad[63:32]** and **p_cbe_l[7:4]** for both address and data phases. Signal **p_par64** is driven by the initiator and is valid one clock cycle after the first address phase when a dual-address command is used and **p_req64_l** is asserted. Signal **p_par64** is also valid one clock cycle after the second address phase of a dual-address transaction when **p_req64_l** is asserted. Signal **p_par64** is valid one clock cycle after valid data is driven (indicated by assertion of **p_irdy_l** for write data and **p_trdy_l** for read data), when both **p_req64_l** and **p_ack64_l** are asserted for that data phase. Signal **p_par64** is tristated by the device driving read or write data one clock cycle after the **p_ad** lines are tristated. Devices receiving data sample **p_par64** as an input to check for possible parity errors during 64-bit transactions. When not driven, **p_par64** is pulled up to a valid logic level through external resistors.
p_req64_l	STS	Primary PCI interface request 64-bit transfer. Signal **p_req64_l** is sampled at secondary reset to enable the 64-bit extension on the primary bus. When sampled low, the 64-bit extension is enabled. Signal **p_req64_l** is asserted by the initiator to indicate that the initiator is requesting 64-bit data transfer. Signal **p_req64_l** has the same timing as **p_frame_l**. When deasserting, **p_req64_l** is driven to a deasserted state for one clock cycle and is then sustained by an external pull-up resistor. The 21555 samples **p_req64_l** during primary bus reset to enable the 64-bit extension signals. When **p_req64_l** is sampled high during reset, the primary 64-bit extension is disabled and assumed not connected. The 21555 then drives **p_ad[63:32]**, **p_cbe_l[7:4]**, and **p_par64** to valid logic levels.

3.3 Secondary PCI Bus Interface Signals

Table 8 describes the secondary PCI bus interface signals. The letters in the "Type" column are described in Table 1.

Table 8. Secondary PCI Bus Interface Signals (Sheet 1 of 2)

Signal Name	Type	Description
s_ad[31:0]	TS	Secondary PCI interface address and data. These signals are a 32-bit multiplexed address and data bus. During the address phase or phases of a transaction, the initiator drives a physical address on **s_ad[31:0]**. During the data phases of a transaction, the initiator drives write data, or the target drives read data, on **s_ad[31:0]**. When the secondary PCI bus is idle, the 21555 drives **s_ad** to a valid logic level when its secondary bus grant is asserted.
s_cbe_l[3:0]	TS	Secondary PCI interface command and byte enables. These signals are a multiplexed command field and byte enable field. During the address phase or phases of a transaction, the initiator drives the transaction type on **s_cbe_l[3:0]**. When there are two address phases, the first address phase carries the dual-address command and the second address phase carries the transaction type. For both read and write transactions, the initiator drives byte enables on **s_cbe_l[3:0]** during the data phases. When the secondary PCI bus is idle, the 21555 drives **s_cbe_l** to a valid logic level when its secondary bus grant is asserted.
s_devsel_l	STS	Secondary PCI interface **DEVSEL#**. Signal **s_devsel_l** is asserted by the target, indicating that the device is responding to the transaction. As a target, the 21555 decodes the address of a transaction initiated on the secondary bus to determine whether to assert **s_devsel_l**. As an initiator of a transaction on the secondary bus, the 21555 looks for the assertion of **s_devsel_l** within five clock cycles of **s_frame_l** assertion; otherwise, the 21555 terminates the transaction with a master abort. Upon completion of a transaction, **s_devsel_l** is driven to a deasserted state for one clock cycle and is then sustained by an external pull-up resistor.
s_frame_l	STS	Secondary PCI interface **FRAME#**. Signal **s_frame_l** is driven by the initiator of a transaction to indicate the beginning and duration of an access on the secondary PCI bus. Signal **s_frame_l** assertion (falling edge) indicates the beginning of a PCI transaction. While **s_frame_l** remains asserted, data transfers can continue. The deassertion of **s_frame_l** indicates the final data phase requested by the initiator. Upon completion of a transaction, **s_frame_l** is driven to a deasserted state for one clock cycle and is then sustained by an external pull-up resistor.
s_idsel	I	Secondary PCI interface **IDSEL**. Signal **s_idsel** is used as the chip select line for Type 0 configuration accesses to 21555 configuration space from the secondary bus. When **s_idsel** is asserted during the address phase of a Type 0 configuration transaction, the 21555 responds to the transaction by asserting **s_devsel_l**.
s_irdy_l	STS	Secondary PCI interface **IRDY#**. Signal **s_irdy_l** is driven by the initiator of a transaction to indicate the initiator's ability to complete the current data phase on the secondary PCI bus. During a write transaction, assertion of **s_irdy_l** indicates that valid write data is being driven on the **s_ad** bus. During a read transaction, assertion of **s_irdy_l** indicates that the initiator is able to accept read data for the current data phase. Once asserted during a given data phase, **s_irdy_l** is not deasserted until the data phase completes. Upon completion of a transaction, **s_irdy_l** is driven to a deasserted state for one clock cycle and is then sustained by an external pull-up resistor.

Table 8. Secondary PCI Bus Interface Signals (Sheet 2 of 2)

Signal Name	Type	Description
s_par	TS	Secondary PCI interface parity. Signal **s_par** carries the even parity of the 36 bits of **s_ad[31:0]** and **s_cbe_l[3:0]** for both address and data phases. Signal **s_par** is driven by the same agent that drives the address (for address parity) or the data (for data parity). Signal **s_par** contains valid parity one clock cycle after the address is valid (indicated by assertion of **s_frame_l**), or one clock cycle after the data is valid (indicated by assertion of **s_irdy_l** for write transactions and **s_trdy_l** for read transactions). Signal **s_par** is tristated one clock cycle after the **s_ad** lines are tristated. The device receiving data samples **s_par** as an input to check for possible parity errors. When the secondary PCI bus is idle, the 21555 drives **s_par** to a valid logic level when its secondary bus grant is asserted (one clock cycle after the **s_ad** bus is parked).
s_stop_l	STS	Secondary PCI interface **STOP#**. Signal **s_stop_l** is driven by the target of a transaction, indicating that the target is requesting the initiator to stop the transaction on the secondary bus. When **s_stop_l** is asserted in conjunction with **s_trdy_l** and **s_devsel_l** assertion, a disconnect with data transfer is being signaled. When **s_stop_l** and **s_devsel_l** are asserted, but **s_trdy_l** is deasserted, a target disconnect without data transfer is being signaled. When this occurs on the first data phase, that is, no data is transferred during the transaction, this is referred to as a target retry. When **s_stop_l** is asserted and **s_devsel_l** is deasserted, the target is signaling a target abort. Upon completion of a transaction, **s_stop_l** is driven to a deasserted state for one clock cycle and is then sustained by an external pull-up resistor.
s_trdy_l	STS	Secondary PCI interface **TRDY#**. Signal **s_trdy_l** is driven by the target of a transaction to indicate the target's ability to complete the current data phase on the secondary PCI bus. During a write transaction, assertion of **s_trdy_l** indicates that the target is able to accept write data for the current data phase. During a read transaction, assertion of **s_trdy_l** indicates that the target is driving valid read data on the **s_ad** bus. Once asserted during a given data phase, **s_trdy_l** is not deasserted until the data phase completes. Upon completion of a transaction, **s_trdy_l** is driven to a deasserted state for one clock cycle and is then sustained by an external pull-up resistor.

3.4 Secondary PCI Bus Interface 64-Bit Extension Signals

Table 9 describes the secondary PCI bus interface 64-bit extension signals. The letters in the “Type” column are described in Table 1.

Table 9. Secondary PCI Bus Interface 64-Bit Extension Signals (Sheet 1 of 2)

Signal Name	Type	Description
s_ack64_l	STS	Secondary PCI interface acknowledge 64-bit transfer. Signal **s_ack64_l** should never be driven when **s_req64_l** is not driven. Signal **s_ack64_l** is asserted by the target only when **s_req64_l** is asserted by the initiator, to indicate the target’s ability to transfer data using 64 bits. Signal **s_ack64_l** has the same timing as **s_devsel_l**. When deasserting, **s_ack64_l** is driven to a deasserted state for one clock cycle and is then sustained by an external pull-up resistor.
s_ad[63:32]	TS	Secondary PCI interface address and data upper 32 bits. The 21555 does not bus park these pins. These pins are tristated during the assertion of **s_rst_l**. Signals **s_ad[63:32]** are driven to a valid value when the 64-bit extension is disabled (**s_req64_l** is deasserted during **s_rst_l** assertion). This multiplexed address and data bus provides an additional 32 bits to the secondary interface. During the address phase or phases of a transaction, when the dual-address command is used and **s_req64_l** is asserted, the initiator drives the upper 32 bits of a 64-bit address; otherwise, these bits are undefined, and the initiator drives a valid logic level onto the pins. During the data phases of a transaction, the initiator drives the upper 32 bits of 64-bit write data, or the target drives the upper 32 bits of 64-bit read data, when **s_req64_l** and **s_ack64_l** are both asserted. When not driven, signals **s_ad[63:32]** are pulled up to a valid logic level through external resistors.
s_cbe_l[7:4]	TS	Secondary PCI interface command and byte enables upper 4 bits. The 21555 does not bus park these pins. These pins are tristated during the assertion of **s_rst_l**. Signals **s_cbe_l[7:4]** are driven to a valid value when the 64-bit extension is disabled (**s_req64_l** is deasserted during **s_rst_l** assertion). These signals are a multiplexed command field and byte enable field. During the address phase or phases of a transaction, when the dual-address command is used and **s_req64_l** is asserted, the initiator drives the transaction type on **s_cbe_l[7:4]**; otherwise, these bits are undefined, and the initiator drives a valid logic level onto the pins. For both read and write transactions, the initiator drives byte enables for the **s_ad[63:32]** data bits on **s_cbe_l[7:4]** during the data phases, when **s_req64_l** and **s_ack64_l** are both asserted. When not driven, signals **s_cbe_l[7:4]** are pulled up to a valid logic level through external resistors.

Table 9. Secondary PCI Bus Interface 64-Bit Extension Signals (Sheet 2 of 2)

Signal Name	Type	Description
s_par64	TS	Secondary PCI interface upper 32 bits parity. The 21555 does not bus park this pin. This pin is tristated during the assertion of **s_rst_l**. Signal **s_par64** is driven to a valid value when the 64-bit extension is disabled (**s_req64_l** is deasserted during **s_rst_l** assertion). Signal **s_par64** carries the even parity of the 36 bits of **s_ad[63:32]** and **s_cbe_l[7:4]** for both address and data phases. Signal **s_par64** is driven by the initiator and is valid one clock cycle after the first address phase when a dual-address command is used and **s_req64_l** is asserted. Signal **s_par64** is also valid one clock cycle after the second address phase of a dual-address transaction when **s_req64_l** is asserted. Signal **s_par64** is valid one clock cycle after valid data is driven (indicated by assertion of **s_irdy_l** for write data and **s_trdy_l** for read data), when both **s_req64_l** and **s_ack64_l** are asserted for that data phase. Signal **s_par64** is tristated by the device driving read or write data one clock cycle after the **s_ad** lines are tristated. Devices receiving data sample **s_par64** as an input to check for possible parity errors during 64-bit transactions. When not driven, **s_par64** is pulled up to a valid logic level through external resistors.
s_req64_l	STS	Secondary PCI interface request 64-bit transfer. Signal **s_req64_l** is sampled at secondary reset to enable the 64-bit extension on the secondary bus. When sampled low, the 64-bit extension is enabled. When designated as a secondary bus central function, the 21555 asserts this signal during secondary bus reset. Signal **s_req64_l** is asserted by the initiator to indicate that the initiator is requesting 64-bit data transfer. Signal **s_req64_l** has the same timing as **s_frame_l**. When the 21555 is the secondary bus central function, it will assert **s_req64_l** low during secondary bus reset to indicate that a 64-bit bus is supported. When deasserting, **s_req64_l** is driven to a deasserted state for one clock cycle and is then sustained by an external pull-up resistor. The 21555 samples **s_req64_l** during secondary bus reset to enable the 64-bit extension signals. When **s_req64_l** is sampled high during reset, the secondary 64-bit extension is disabled and assumed not connected. The 21555 then drives **s_ad[63:32]**, **s_cbe_l[7:4]**, and **s_par64** to valid logic levels.

3.5 Miscellaneous Signals

Table 10 describes the miscellaneous signals. The letters in the “Type” column are described in Table 1.

Table 10. Miscellaneous Signals

Signal Name	Type	Description
p_vio	I	Primary interface I/O voltage. This signal must be tied to either 3.3 V or 5.0 V, corresponding to the signaling environment of the primary PCI bus as described in the *PCI Local Bus Specification, Revision 2.2*. When any device on the primary PCI bus uses 5-V signaling levels, tie **p_vio** to 5.0 V. Signal **p_vio** is tied to 3.3 V only when all the devices on the primary bus use 3.3-V signaling levels.
s_vio	I	Secondary interface I/O voltage. This signal must be tied to either 3.3 V or 5.0 V, corresponding to the signaling environment of the secondary PCI bus as described in the *PCI Local Bus Specification, Revision 2.2*. When any device on the secondary PCI bus uses 5-V signaling levels, tie **s_vio** to 5.0 V. Signal **s_vio** is tied to 3.3 V only when all the devices on the secondary bus use 3.3-V signaling levels.

Address Decoding 4

This chapter presents the theory of operation information about address mapping and decoding. See Chapter 16 for specific information about addressing registers. The following areas are covered:

The 21555 implements separate Base Address Registers (BARs) on both the primary and secondary interfaces. The BARs denotes address ranges for downstream and upstream forwarding. This addressing is unlike the transparent PCI-to-PCI Bridge (PPB), discussed in Chapter 2, which implements a flat address map encompassing both the primary and secondary interfaces.

The 21555 BARs denote interface ranges for Control and Status Registers (CSR) access.

- Primary Interface – The 21555 responds to those transactions whose addresses fall into one of its primary BAR ranges. All other I/O and memory transactions on the primary bus are ignored by the 21555. The address ranges defined by the primary BARs reside in the primary, or system, address map.
- Secondary Interface – The 21555 responds to those transactions whose addresses reside in one of the secondary BAR ranges. All other transactions on the secondary bus are ignored by the 21555. The address ranges defined by the secondary BARs reside in the secondary, or local, address map.

The system and local address maps are independent of each other. The 21555 supports address translations between the two address maps when forwarding transactions upstream or downstream.

> ***Note:*** When enabled as a target, the 21555 ignores any transactions that it initiates as a master with the exception of type 0 configuration transactions.

4.1 CSR Address Decoding

The 21555 implements a set of CSRs that are mapped in memory or in I/O space. The registers are mapped independently on the primary and secondary interfaces. The following BARs are used for CSR mapping:

- The primary CSR and:
  - Downstream Memory 0 BAR is for mapping in primary bus memory address space. The Lower 4KB of this range used to map the 21555 CSRs.
  - I/O BAR is for mapping in primary bus I/O space.
- The secondary CSR:
  - Memory BAR is for mapping in secondary bus memory address space.
  - I/O BAR is for mapping in secondary bus I/O space.

The primary BARs are located in the 21555 primary bus configuration space, and the secondary BARs are located in the 21555 secondary bus configuration space. The memory BARs request 4 KB each (minimum size for Primary CSR and Downstream Memory 0 BAR), and the I/O BARs request 256 bytes each.

4.2 Expansion ROM Address Mapping (Decoding)

The 21555 implements one BAR, the Primary Expansion Read Only Memory (ROM) BAR, to map the expansion ROM that can be attached to the 21555. The Expansion ROM can be mapped into primary bus address space only, and is not accessible through a BAR from the secondary bus. The size of the Primary Expansion ROM BAR is programmable through the Primary Expansion ROM Setup register in device-specific configuration space. The size may vary from 4 KB to 16 MB by powers of 2. The Primary Expansion ROM BAR can also be disabled through theSetup register so that it does not request space when the expansion ROM is not implemented.

4.3 Memory 0 Transaction Address Decoding

The BARs can be enabled to decode and forward memory transactions to the opposite interface. The 21555 implements primary interface and secondary interface BARs:

- The downstream BARs are in primary configuration space. The BARs decode transactions on the primary bus to be forwarded to the secondary bus.
  - Primary CSR and Downstream Memory 0.
  - For addresses above the low 4 KB in this address range.
  - Upstream I/O or Memory 0.
  - Upstream Memory 1.
  - Upstream Memory 2.
- The upstream BARs are in secondary configuration space. The BARs decode transactions on the secondary bus to be forwarded to the primary bus.
  - Downstream I/O or Memory 1.
  - Downstream Memory 2.
  - Downstream Memory 3.

4.3.1 Using the BAR Setup Registers

All downstream and upstream BARs have programmable sizes, and can be disabled so that they request no space. The Primary CSR and Downstream Memory 0 BAR cannot be totally disabled, as the 21555 CSRs are always mapped in the bottom 4KB. The forwarding part of the range can be disabled by requesting only 4KB of memory Table 12 on page 47 summarizes the minimum and maximum range for each address range). In addition, the Downstream Memory 3 BAR can be configured to be mapped in 64-bit address space. The register then comprises two 32-bit registers and can be used for forwarding DACs downstream. 64-bit addressing support is discussed further in Section 4.3.5 on page 41. These BARs can also be programmed to be prefetchable or non-prefetchable.

Programming of all the forwarding BARs with the exception of the Upstream Memory 2 BAR is done through corresponding device-specific Setup configuration registers. The Primary Expansion ROM BAR also has a Setup register. Setup registers are preloaded by the serial ROM and can also be written from the secondary interface. Each bit of the Setup register corresponds to the same bit of its respective BAR.

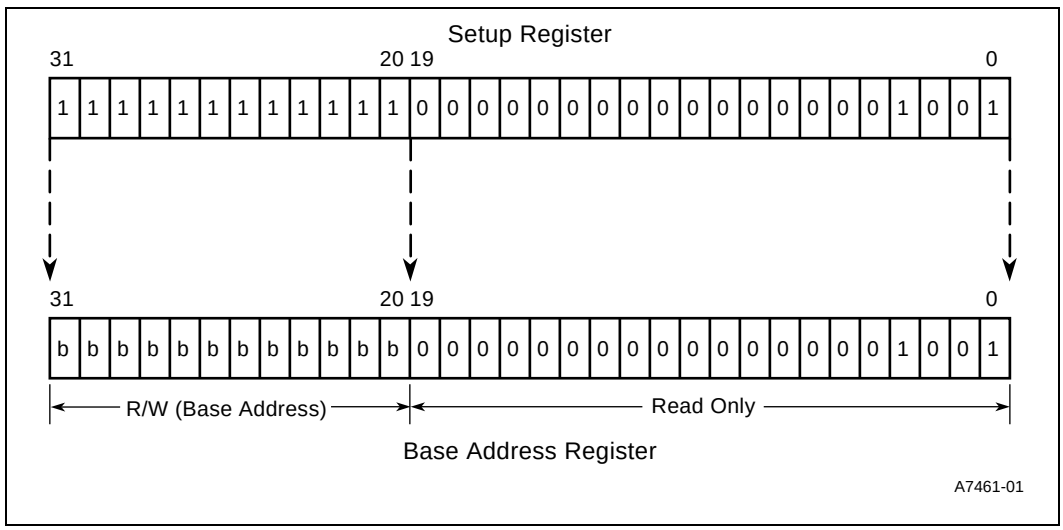
Bit 0 of the Downstream I/O or Memory 1 Setup register and Upstream I/O or Memory 0 Setup register (see Section 16 on page 121) should be written with:

- A zero (0) to select a memory BAR.
- A one (1) to select an I/O BAR. Bits [2:1] are writable to select the type of memory mapping. The Downstream Memory 3 Setup registers bits [2:1] may be set to 10b to select 64-bit addressing.

A mask is used to set the size of the BAR for the remaining read/write bits of the Setup register. Writing a 1 sets the corresponding bit in that Setup register's BAR to be read/write. Writing a 0 (zero) sets the corresponding bit in that Setup register's BAR to be read only as 0. Therefore, the size is set by writing the appropriate number of most significant bits to a 1, and the remaining bits to a 0. When all of the zeros and ones in the size field are not contiguous, this is illegal and unpredictable results may occur. When the most significant writable bit of the Setup register is a 0, the corresponding BAR is disabled and requests no space.

Figure 3 shows an example of using a Setup register to program a BAR to request 1 MB of memory space.

Figure 3. BAR Setup Register Example

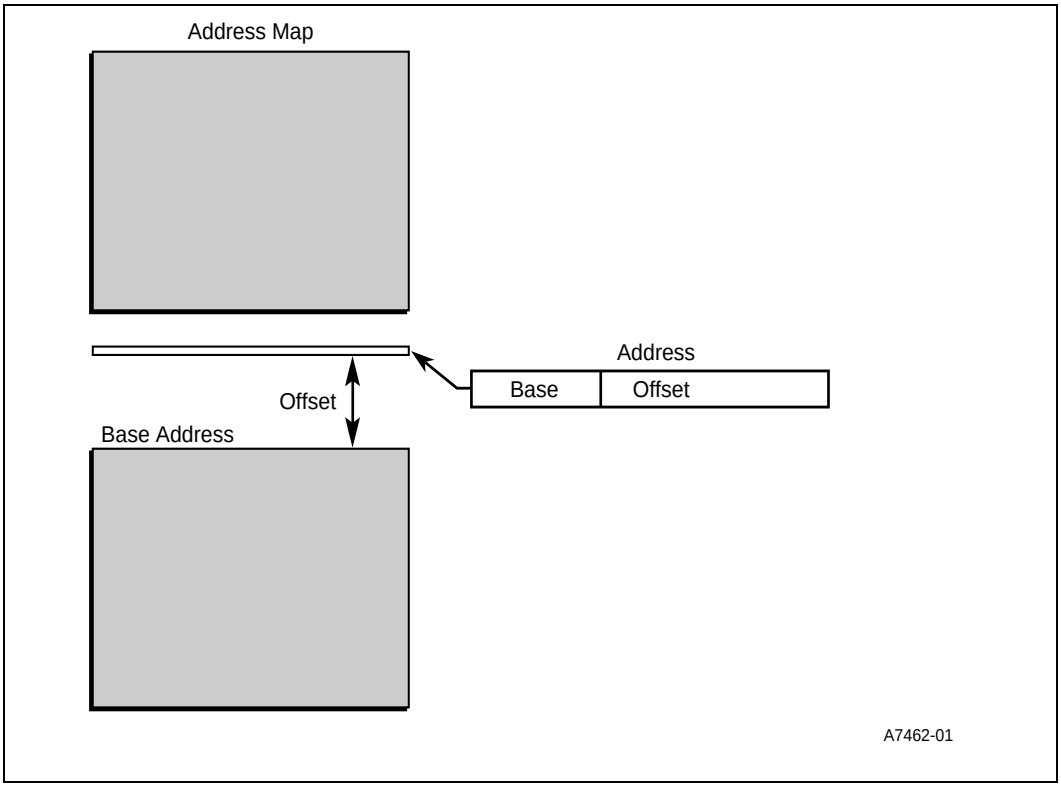


4.3.2 Direct Address Translation

With the exception of secondary bus transactions falling into the Upstream Memory 2 address range (see Section 4.3.3) and all dual address transactions (Section 4.3.5), the 21555 uses direct address translation when forwarding memory transactions from one interface to the other. Note that since transactions addressing the bottom 4 KB of the Primary CSR and Downstream Memory 0 BAR are targeted at the 21555 CSRs, no forwarding and therefore no address translation is performed. Direct address translation is used for transactions in that range above the low 4 KB boundary.

A memory address may be thought of as a base address (as programmed in the Downstream and Upstream BARs) with an offset from the base address, as shown in Figure 4.

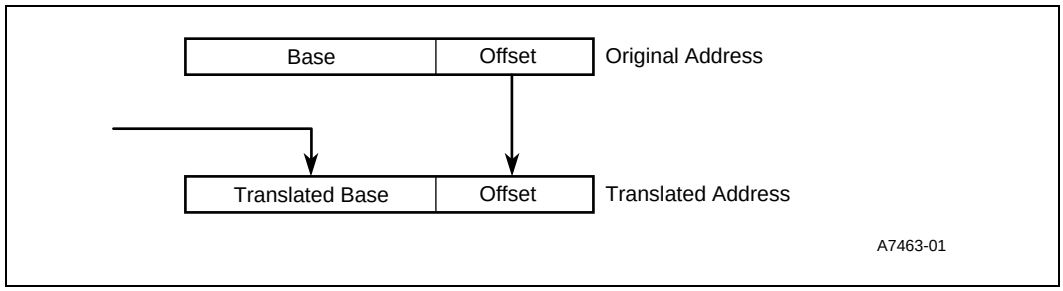
Figure 4. Address Format



When a memory transaction is forwarded downstream from the primary bus to the secondary bus, the primary bus address can be mapped to another address in the secondary bus domain. The mapping is performed by substituting a new base address for the base of the original address, as shown in Figure 5.

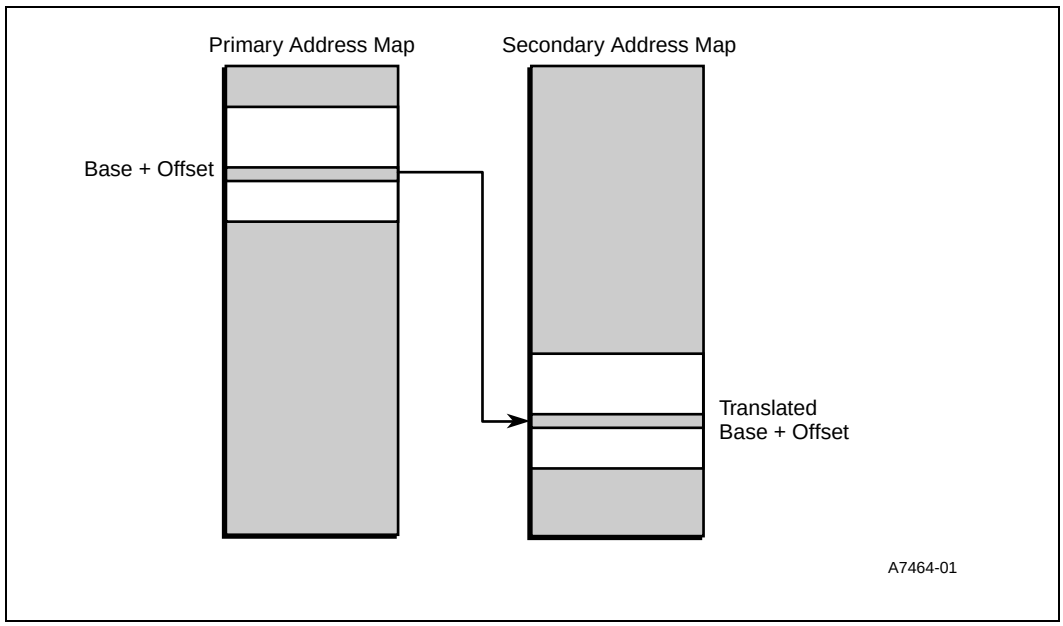
This new base address, also called the translated base address, references a new location in the secondary bus address map. The offset is not affected. The process is similar for transactions forwarded from the secondary bus to the primary bus.

Figure 5. Direct Offset Address Translation



Each memory address range using direct offset address translation has its own translated base. The translated base addresses are programmable in registers corresponding to each BAR. These registers are mapped both in device-specific configuration space and in CSR space. The number of bits of the translated base address corresponds to the number of writable bits in the respective BAR. Likewise, the number of bits of the offset also varies and depends on the size of the BAR. Figure 6 shows an example of address translation of downstream memory transactions. Again, upstream transactions are treated similarly.

Figure 6. Downstream Address Translation Example



4.3.3 Lookup Table Based Address Translation

As mentioned in Section 4.3.2, Upstream Memory 2 address translation is treated differently than the other ranges. The 21555 uses a page size based lookup table to perform address translation for transactions falling into this range. A lookup table provides a flexible way of translating secondary bus local addresses to primary bus system addresses.

The Upstream Memory 2 address range consists of a fixed number (64) of pages. The page size is programmable in the Chip Control 1 configuration register. Therefore, the size of the Upstream Memory 2 BAR is dependent on the page size. The page size varies between 256 bytes to 32 MB by powers of 2. This results in a window size that varies from 16 KB to 2 GB. This BAR can also be disabled.

Each page of the upstream window has a corresponding translated base address.

The size of the translated base address varies with the page size and window size. The translated base address replaces both the original base address and the lookup table index bits. The address bits used for the original base address for a given page and window size are listed in Table 11, as well as the locations of the six address bits needed to select one of the 64 entries in the lookup table. The offset of the address, which is not translated, consists of the remaining lower order address bits. Table 11 shows the Upstream Memory 2 window size, with base address, index, and offset fields.

Table 11. Upstream Memory 2 Window Size

Page Size (bytes)	Window Size (bytes)	Base Address (bits)	Lookup Table Index (bits)	Offset (bits)
256	16K	[31:14]	[13:8]	[7:0]
512	32K	[31:15]	[14:9]	[8:0]
1K	64K	[31:16]	[15:10]	[9:0]
2K	128K	[31:17]	[16:11]	[10:0]
4K	256K	[31:18]	[17:12]	[11:0]
8K	512K	[31:19]	[18:13]	[12:0]
16K	1M	[31:20]	[19:14]	[13:0]
32K	2M	[31:21]	[20:15]	[14:0]
64K	4M	[31:22]	[21:16]	[15:0]
128K	8M	[31:23]	[22:17]	[16:0]
256K	16M	[31:24]	[23:18]	[17:0]
512K	32M	[31:25]	[24:19]	[18:0]
1M	64M	[31:26]	[25:20]	[19:0]
2M	128M	[31:27]	[26:21]	[20:0]
4M	256M	[31:28]	[27:22]	[21:0]
8M	512M	[31:29]	[28:23]	[22:0]
16M	1G	[31:30]	[29:24]	[23:0]
32	2G	[31]	[30:25]	[24:0]

Figure 7 shows how a translated address is built using the lookup table, assuming a page size of 4 KB.

Figure 7. Address Translation Using A Lookup Table

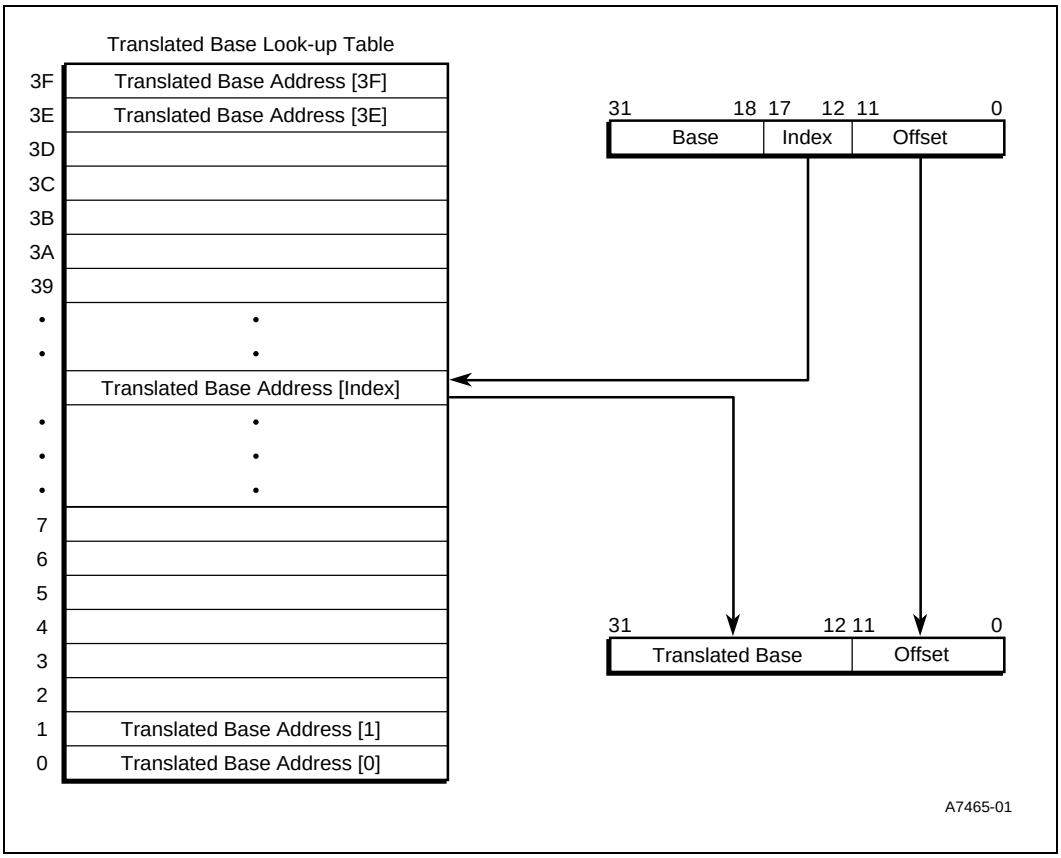
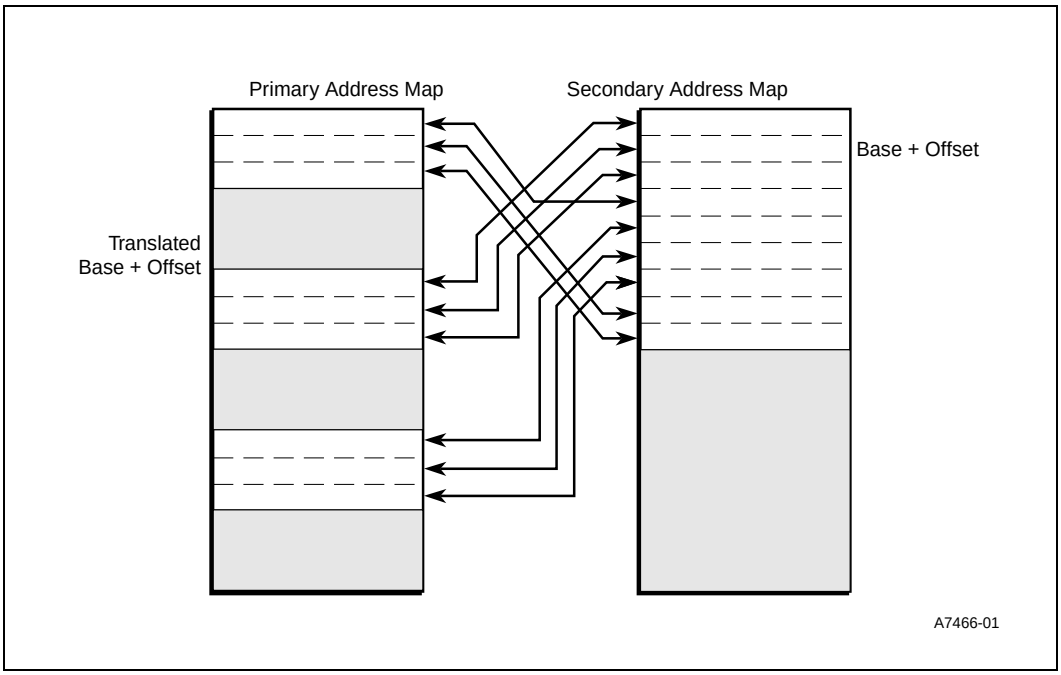


Figure 8 shows an example of how different address regions might be forwarded upstream using the lookup table address translation.

The lookup table is implemented on-chip and no external memory is needed. The lookup table is part of the memory space that the 21555 requests with its Primary CSR Memory BAR and Secondary CSR Memory BAR. The lookup table is also indirectly accessible in I/O or memory space at offsets 24h and 28h.

Note: The indirect access mechanism must be used only by one interface at a time. When access to the lookup table by multiple masters is possible, it is strongly recommended that the Generic Own bits or some other semaphore mechanism be used to restrict access to one master at a time.

Figure 8. Upstream Lookup Table Address Translation



The 21555 conditionally asserts **s_inta_l** when an upstream memory transaction transfers data addressing the last Dword in a page. This interrupt alerts the local processor that the page entry may need updating. The 21555 implements an event bit and interrupt mask bit for each of the 64 pages (entries) in the upstream window.

Note: The page entry of the lookup table should not be updated while the initiator is still performing transactions addressing that page.

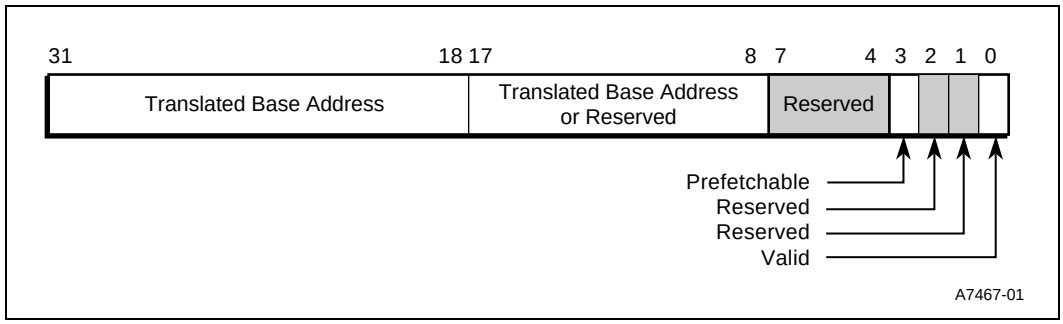
4.3.4 Lookup Table Entry Format

Figure 9 shows the format for an entry in the lookup table. The number of bits of the entry used for the new translated base address is variable and is listed in Table 11. The maximum number of bits used are bits [31:8], corresponding to a 256-byte page size, while the minimum number of bits used is bit [31], corresponding to a 32 Mbyte page size. The next 4 to 27 bits, depending on the number of bits used for the base address, are reserved. The low 4 bits are used for control. Two control bits are defined, one indicating whether the entry is a valid entry, and one indicating whether prefetchable behavior should be used on memory reads. When the entry is not valid, the 21555 treats the transaction addressing that page as if a master abort were detected on the target interface.

For writes, the 21555 discards memory write data and asserts **s_serr_l**, when the **SERR#** Disable for Master Abort during Posted Write bit is 0. For reads, the 21555 returns FFFFFFFFh on reads if the Master Abort Mode bit is 0, or returns a target abort if the Master Abort Mode bit is a 1.

Note: The lookup table is not cleared by reset. The lookup table must be initialized by the local processor before the Upstream Memory 2 Address range is used.

Figure 9. Lookup Table Entry Format



4.3.5 Forwarding of 64-Bit Address Memory Transactions

The 21555 considers the host and local memory space above the 4 GB boundary to be shared. This means that the 21555 uses a flat address map in this space. Dual-address cycle (DAC) transactions are used for addressing above the 4 GB boundary. The 21555 can forward dual-address cycle transactions both upstream and downstream. The Downstream Memory 3 BAR is used to designate the address range for downstream DACs. Inverse decoding is used for upstream DACs.

The Downstream Memory 3 BAR may be configured to be a 64-bit BAR by preloading the Downstream Memory 3 Upper 32 Bits Setup register bit [31] to a one. The Downstream Memory 3 Setup register bits [2:1] should be set to 10b. This implies that the memory range can be located anywhere in 64-bit address space. When this 64-bit addressing option is used, the maximum window size changes from 2 GB (in the 32-bit case) to 2^{63} bytes.

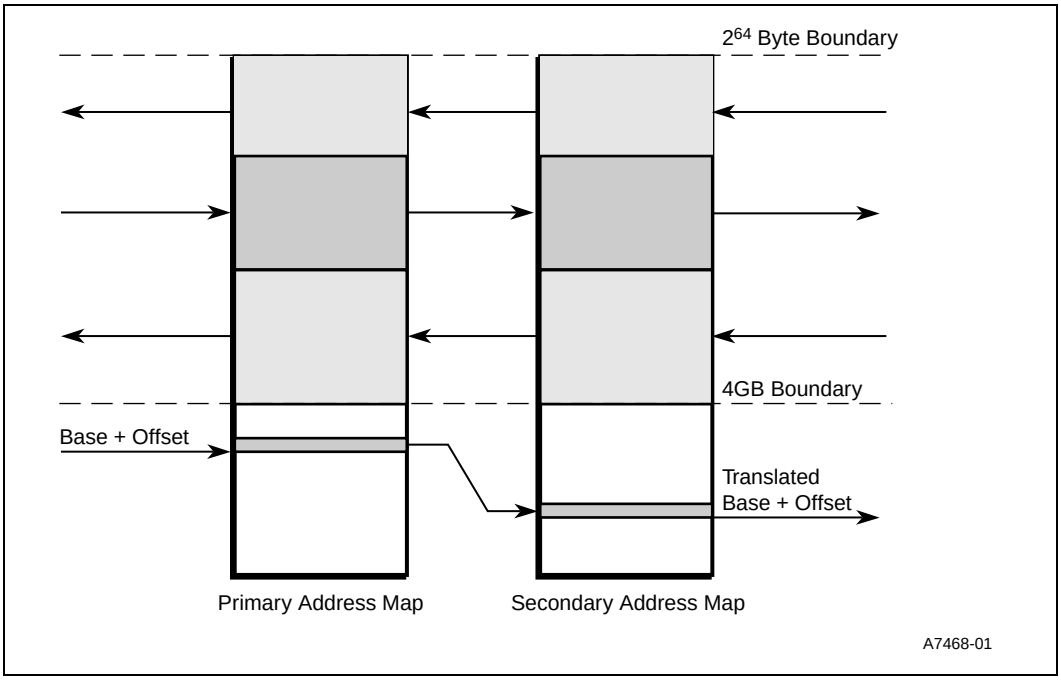
When the preloaded window size for a 64-bit BAR is 2 GB or less, the space requested may be mapped either in 32-bit address space or 64-bit address space. In the former case, the upper 32 bits of the base address is zero and transactions are forwarded as described in the previous section using direct offset address translation. When the upper 32-bit base address is non-zero, the memory range is located above the 4 GB boundary.

When the Downstream Memory 3 Range is mapped above the 4 GB boundary, primary bus transactions falling into this address range are forwarded downstream with no address translation performed. Any 64-bit address transactions on the secondary bus falling outside of the Downstream Memory 3 address range are forwarded upstream, again with no address translation. This is similar to the forwarding mechanisms of a transparent PPB (21154) and is illustrated in Figure 10.

Note: Since the use of BARs restricts the alignment of the address range to the window size, the Downstream Memory 3 address range can never straddle the 4 GB boundary. The base address of

the Downstream memory 3 address range must be set to a non-zero value when the upper 32 bits are enabled (a base address of 0 is not allowed).

Figure 10. Dual-Address Transaction Forwarding



4.4 I/O Transaction Address Decoding

The 21555 provides a mechanism where one BAR on each interface can be configured to be an I/O BAR instead of a memory BAR. The Downstream I/O or Memory 1 BAR in primary configuration space is used to decode primary bus I/O transactions for forwarding to the secondary bus. The Upstream I/O or Memory 0 BAR in secondary configuration space is used to decode secondary bus I/O transactions for forwarding to the primary bus. (See Table 37 on page 133.)

The 21555 performs direct offset address translation when forwarding I/O transactions in much the same manner that it translates memory addresses. The size of the I/O BARs can be configured to be 64 bytes, 128 bytes, or 256 bytes. Accordingly, the base address can consist of 26, 25, or 24 bits. The 21555 hardware does not restrict setting up larger I/O windows, although requesting more than 256 bytes of I/O space is a violation of the *PCI Local Bus Specification, Revision 2.2.* The upper bits comprising the base address of the I/O address on the primary bus is replaced with the base address written in the Downstream I/O or Memory 1 Translated BAR when initiated on the secondary bus. Similarly, the Upstream I/O or Memory 0 Translated BAR is used for upstream I/O transactions. These translated base registers are mapped in both device-specific configuration space and the 21555 CSR space.

4.4.1 Indirect I/O Transaction Generation

The 21555 implements a CSR mechanism that allows access to any I/O address in the secondary or local I/O address map from the primary interface, or any I/O address in the primary or host I/O address map from the secondary interface. A pair of device-specific CSR registers contain the address and data used to initiate the I/O

transaction. One pair is used for downstream I/O transactions and one pair is used for upstream I/O transactions. The downstream registers can only be accessed from the primary interface, and the upstream registers can only be accessed from the secondary interface. Their function is similar, so only the downstream case is discussed.

The Downstream I/O Address register contains the address used when the transaction is initiated on the secondary bus. When the Downstream I/O Data register is read or written from the primary interface, the 21555 initiates the transaction on the secondary bus. For writes, the Downstream I/O Data register contains the write data to be written. For reads, the read data is placed in this register upon completion of the secondary bus I/O read.

The I/O Data register must be accessed with an I/O transaction on the primary interface to initiate the secondary bus I/O transaction. Otherwise, this register appears as reserved for both memory accesses or accesses from the secondary interface. The Downstream I/O Control bit in the I/O CSR must be set to enable downstream I/O transaction generation; otherwise, I/O Data register accesses are treated as reserved accesses.

The 21555 uses the same byte enables that the initiator used to read or write the register.

> ***Note:*** The low bits of the I/O address in the I/O Address register must match the byte enables as described in the *PCI Local Bus Specification, Revision 2.2*. The 21555 will not correct discrepancies between byte enables and address bits [1:0].

The 21555 responds to read or write access of Downstream I/O Data register with a target retry until the access is completed on the secondary bus. This I/O access is treated as a delayed transaction by the 21555. This delayed transaction is entered into the 21555's downstream delayed transaction queue and is ordered with respect to all other downstream transactions. When ordering rules permit, the 21555 initiates I/O write or read on the secondary bus. When the I/O transaction completes, the 21555 returns target termination and, if a read, returns read data when the initiator repeats the transaction.

The 21555 provides a semaphore method that may be used to guarantee atomicity of the Downstream I/O Address and Downstream I/O Data register accesses using the Downstream I/O Own bit. Atomicity of these accesses is not hardware-enforced. An Upstream I/O Own bit is provided for upstream I/O transactions. The following procedure should be used for downstream
I/O transactions:

1. The initiator of the transaction reads the Downstream I/O Own bit. When the bit reads as zero, the initiator can proceed with the indirect I/O transaction sequence. When the bit reads as a 1, the initiator should not proceed until a subsequent read of the own bit returns a 0 (zero). The 21555 automatically sets the own bit to a 1after it is read from the primary interface.
2. The initiator writes the target I/O address in the Downstream I/O Address register.
3. The initiator should write or read the data in the Downstream I/O Data register until a response other than target retry is received.
4. Upon returning the completion of the I/O transaction to the initiator, the 21555 automatically clears the bit to a 0.

The same procedure should be used for upstream I/O transactions using the Upstream I/O Address register, Upstream I/O Data register, and Upstream I/O Own bit. To read the state of the Downstream and Upstream I/O Own bits without side effects, a read-only copy of the I/O Own bit states is kept in the I/O CSR. Byte access of the I/O Own bits and their
read-only copies should be used to avoid setting the I/O Own bit for the opposite interface.

4.4.2 Subtractive Decoding of I/O Transactions

The 21555 can be enabled to subtractively decode I/O transactions and forward these transactions to the opposite bus. No address translation is performed on subtractively decoded I/O transactions. The transaction is treated by the 21555 as a delayed transaction.

> ***Note:*** Even when a subtractively-decoded delayed transaction is queued, the 21555 continues to respond to the transaction on the initiator bus with subtractive timing.

To enable subtractive decoding of I/O transactions on the primary bus, the Subtractive Decode Enable bits in the Chip Control 1 configuration register must be set to 01b. To enable subtractive decoding of I/O transactions on the secondary bus, the Subtractive Decode Enable bits must be set to 10b.

There can be only one subtractive decoding agent on a PCI bus. Subtractive decoding should not be enabled for both the primary and secondary interfaces.

4.5 Configuration Accesses

The 21555 responds to Type 0 configuration transactions on both its primary and secondary interfaces. The 21555 has two sets of configuration registers, one for the primary interface and one for the secondary interface. Both sets are accessible from either interface. The 21555 can act as an initiator of Type 0 or Type 1 configuration transactions on the primary or secondary bus using the indirect configuration transaction mechanism.

4.5.1 Type 0 Accesses to 21555 Configuration Space

The 21555 responds as a target to Type 0 configuration transactions on both its primary and secondary interfaces when the IDSEL pin for that interface is asserted. The 21555 is a single function device so it does not decode the function number. The 21555 can respond to configuration transactions regardless of the state of the posted write and delayed transaction queues. Because the 21555 is not a transparent PPB (21154), it does not respond to Type 1 configuration transactions.

Access to the 21555 configuration space may be restricted during different phases of initialization:

- Reset:

  No access to the 21555 configuration space from either interface.
- Serial preload:

  No access to the 21555 configuration space from either interface. the 21555 returns target retry.
- Optional primary lockout:

  Access to the 21555 configuration space is allowed from the secondary interface only, until the Primary Lockout Bit in the Chip Control 0 register is cleared. The 21555 returns target retry to all accesses initiated on the primary bus, with the exception of accesses to the Table 123, “Reset Control Register” on page 188 at Dword D8h.
- Normal configuration and operation:

  Access to the 21555 configuration space is allowed from both the primary and secondary interfaces.

See Chapter 2 for a more detailed description of the initialization process.

During normal configuration and operation, when the 21555 decodes a configuration access on one interface while an access to a configuration register is already ongoing on the other interface, the 21555 holds the second initiator in wait states until the first transaction is complete, and then completes the second transaction.

Accesses to the 21555 configuration space are not ordered with respect to transactions in the 21555 queues. That is, the 21555 responds immediately to configuration transactions regardless of what transactions exist in the upstream and downstream queues. Exceptions to this are configuration accesses that result in the initiation of configuration and I/O transactions by the 21555. These transactions are entered in the delayed transaction queue and ordered appropriately with respect to other delayed transactions and posted writes in the 21555 queues.

4.5.2 Initiation of Configuration Transactions by 21555

Usually, the host processor configures primary bus devices and the local processor configures secondary bus devices, so forwarding of configuration transactions is not typically necessary. However, to support other configuration methods, the 21555 implements a mechanism that enables initiation of Type 0 or Type 1 configuration accesses on either the primary bus or the secondary bus. This mechanism is different from the hierarchical mechanisms supported by PPBs. Instead, two pairs of device-specific registers contain the address and data that are used to initiate the configuration transaction. One pair is used to generate transactions on the primary interface; the other is used to generate transactions on the secondary interface:

- The Upstream Configuration Address and Upstream Configuration Data registers contain the address and data of the configuration transaction to be initiated on the primary bus.
- the Downstream Configuration Address and Downstream Configuration Data registers contain the address and data of the configuration transaction to be initiated on the secondary bus.

In addition, the Configuration CSR and Configuration Own Bits register are used for configuration transaction generation. All of these registers are mapped into both device specific configuration space and the 21555 CSR space. The upstream address and data registers can be written from the secondary interface only, and the downstream address and data registers can be written from the primary interface only. Downstream and upstream configuration address registers can be read from either interface. Otherwise, these registers respond as reserved.

To generate a configuration transaction, the corresponding Upstream or Downstream Configuration Control bit in the Configuration CSR must be set. Otherwise, the corresponding Configuration Data registers are treated as reserved registers. The Configuration Data registers are also treated as reserved registers in memory space.

The Upstream or Downstream Configuration Address register must be written with the address to be driven before the corresponding data register is accessed. This address is driven on the AD lines exactly as written in the register. Therefore, a Type 0 format must be used to generate a
Type 0 configuration transaction, and a Type 1 format must be used to generate a Type 1 configuration transaction. The upper 21 bits of a Type 0 address format are used as IDSEL signals and are specific to the motherboard or add-in card application.

The configuration transaction is initiated by the 21555 when the Upstream or Downstream Configuration Data register is either read or written from the secondary or primary interface, respectively. These registers must be accessed by either a configuration transaction or an I/O transaction to initiate the transaction. The 21555 uses the same byte enables that the initiator used to read or write the register. The 21555 responds to the access of the Upstream or Downstream Configuration Data register with a target retry until the access is completed on the target bus. When the access is completed, the 21555 returns the corresponding target termination and, if a read, the read data on a subsequent attempt of the transaction by the initiator. When the Delayed Transaction Target Retry Counter expires, that is, 2^{24} target retries are received from the target, the 21555 returns a target abort to the initiator. The Delayed Transaction Target Retry Counter may be disabled, and thus does not limit the number of retries, by setting the Retry Counter Disable bit in the Chip Control 0 configuration register.

The 21555 can be enabled to respond to configuration transactions that it generates by setting the appropriate Downstream/Upstream Self-Response Enable bit in the Configuration CSR. For the 21555 to respond, the transaction must assert the 21555's **IDSEL** signal on that interface, and it must be a Type 0 configuration transaction. When this bit is not set, the 21555 will not respond to any configuration transactions that it generates, and these transactions may end in master abort.

The 21555 provides a semaphore method that may be used to guarantee atomicity of the address and data register accesses using the Upstream Configuration Own bit and Downstream Configuration Own bit. Atomicity of these accesses is not guaranteed in hardware. When the corresponding Configuration Enable bit is not set, the Own bit is treated as reserved. The following procedure should be used for downstream transactions:

1. The initiator of the transaction should read the Downstream Configuration Own bit for initiation of transactions on the secondary bus. When the bit reads as zero, the initiator may proceed with the configuration transaction sequence. When the bit reads as a one, the initiator should not proceed until a subsequent read of the own bit returns a 0 (zero). The 21555 automatically sets the own bit to a 1 after it is read.
2. The initiator should write the target configuration address in the Downstream Configuration Address register.
3. The initiator should write or read the data in the Downstream Configuration Data register until a response other than target retry is received.
4. Upon completion of the configuration transaction on the initiator bus, the 21555 automatically clears the Downstream Configuration Own bit to a 0.

Upstream configuration transactions should use a similar process. To check the status of the own bits without read side effects, read only copies of these bits are located in the Configuration CSR. Byte access of the Configuration Own bits and their read-only copies should be used to avoid setting the Configuration Own bit for the opposite interface.

4.6 21555 Bar Summary

Table 12 shows a summary of the 21555 BARs.

Table 12. Bar Summary

Bar	Size	Address Translation
Primary CSR and Downstream Memory 0	4 KB to 2 GB	Low 4 KB: None Above 4KB boundary: Direct Offset
Primary CSR I/O	256 bytes	—
Secondary CSR Memory	4 KB	—
Secondary CSR I/O	256 bytes	—
Primary Expansion ROM	4 KB to 16 MB	—
Downstream I/O or Memory 1	64 bytes to 256 bytes (I/O) or 4 KB to 2 GB (memory)	Direct Offset
Downstream Memory 2	4 KB to 2 GB	Direct Offset
Downstream Memory 3	4 KB to 2^{63} bytes	Direct Offset (< 4 GB) None (Š4 GB)
Upstream I/O or Memory 0	64 bytes to 256 bytes (I/O) or 4 KB to 2 GB (memory)	Direct Offset
Upstream Memory 1	4 KB to 2 GB	Direct Offset
Upstream Memory 2	64 KB to 16 MB	Lookup Table

PCI Bus Transactions 5

This chapter presents the theory of operation information about PCI transactions. See Chapter 16 for specific information about PCI registers. The following sections are discussed:

5.1 Transactions Overview

The 21555 responds to transactions using these commands as a target on both interfaces. The 21555 does not respond to transactions using any other PCI commands.

- All memory commands.
- Dual-address commands.
- I/O read and write commands.
- Type 0 configuration commands.

The 21555 can initiate transactions using the Type 0 and Type 1configuration commands on either interface.

The 21555:

- Responds to transactions by asserting **DEVSEL#** with medium timing.
- Can subtractively decode I/O transactions in the primary direction only.
- Supports linear increment address mode only and disconnects memory transactions whose low two address bits are not 00b after a single Dword.

5.2 Posted Write Transactions

This section discusses the following Posted Write Transactions:

- Section 5.2.1, “Memory Write Transactions” on page 51.
- Section 5.2.2, “Memory Write and Invalidate Transactions” on page 51.
- Section 5.2.3, “64-bit Extension Posted Write Transaction” on page 52.
- Section 5.2.4, “Write Performance Tuning Options” on page 52.

The 21555 posts all memory write and Memory Write and Invalidate (MWI) transactions that are to be forwarded from one interface to the other. The 21555 accepts write data into its buffers without wait states until one of the following conditions occur:

- The initiator ends the transaction.
- An aligned address boundary is reached.
- The posted write queue fills.

Aligned address disconnect boundaries for memory write and MWI transactions are listed in Section 5.2.1 and Section 5.2.2.

The 21555 does not initiate a memory write transaction on the target bus until at least a cache line amount of data is posted. When the transaction consists of less than a cache line, the 21555 waits until the entire burst is posted. For all posted write behavior dependent on the cache line size (CLS), the 21555 uses the cache line value corresponding to the target interface. For downstream transactions the secondary bus cache line size is used, and for upstream transactions the primary cache line size is used. When the cache line size corresponding to the target bus is not set to a valid value, the 21555 uses a value of 8 Dwords for this purpose. Possible valid values are 8, 16 and 32 Dwords.

Note: A cache line amount of data refers to the number of Dwords only, no address alignment is inferred.

The 21555 continues the transaction to the target as long as write data is available or the transaction has terminated on the initiator bus. Otherwise, the 21555 ends the transaction when a queue-empty condition is detected or when all write data has been delivered for this transaction. The 21555 does not insert master wait states when initiating posted writes.

Note: A queue empty condition occurs when less than a cache line amount of data exists in the posted write buffers. This does not imply any address alignment; in this context cache line refers only to the number of Dwords, and the transaction is not necessarily ended on a cache line boundary.

When the 21555 receives 2^{24} consecutive target retries from the target when attempting to deliver posted write data, the 21555 discards the posted write transaction and conditionally asserts **SERR#** on the initiator bus (see Chapter 12). This retry counter can be disabled by setting the retry counter disable bit in the Chip Control 0 configuration register. The 21555 also conditionally asserts **SERR#** on the initiator bus when a target abort or master abort is detected on the target bus in response to the posted write.

5.2.1 Memory Write Transactions

As a target, the 21555 disconnects memory write transactions at the following address boundaries:

- An aligned 4KB address boundary.
- An aligned page address boundary for upstream transactions falling in the Upstream Memory 2 address range.
- An aligned cache line boundary, when the MW disconnect bit is set in configuration space.

When the posted write queue fills before the master terminates the transaction, the 21555 returns a target disconnect when the last queue entry is filled. The 21555 does not disconnect on an aligned address boundary, other than those noted in the previous paragraph, when the write queue is almost full. That is, the memory write queue full disconnect condition is optimized for burst length and not alignment.

As an initiator, when the 21555 has posted write data to deliver and the conditions listed in Section 5.2.2 for initiating an MWI transaction are not met, the 21555 uses the memory write command to deliver posted memory write data. The 21555 terminates the memory write burst when the last piece of data in the transaction is delivered, or if the transaction is in flow-through mode, when a queue empty condition is detected. In the latter case, the 21555 master terminates the transaction on the target bus, and then initiates a new transaction when a cache line amount of data is accumulated.

5.2.2 Memory Write and Invalidate Transactions

As a target, the 21555 disconnects MWI transactions at the following address boundaries:

- An aligned 4 KB address boundary.
- An aligned page address boundary, for upstream transactions falling in the Upstream Memory 2 address range.
- An aligned cache line boundary, for MWI transactions when less than a cache line of available space remains in the posted write queue.

The 21555 disconnects an MWI on a cache line boundary when less than a cache line remains free in the posted write buffer. This is a different queue full disconnect behavior than that used for the memory write command. In this case, alignment is preserved at the expense of maximizing burst length.

When a master initiates an MWI transaction, it guarantees that it will supply one full cache line of data, or some multiple thereof. The 21555 initiates an MWI transaction on the target bus, regardless of whether the bus command was a memory write or an MWI on the initiator bus, when all of the following conditions are met:

- The MWI Enable bit is set in the Command register corresponding to the target interface.
- The target bus Cache Line Size is set to a valid value (8, 16, or 32 Dwords).
- At least one aligned cache line of data has been posted.
- All byte enables for the posted cache line are turned on.

When any of these conditions is not met, the 21555 uses the memory write command. When a subsequent cache line in the transaction does not have all bytes enabled, the 21555 terminates the MWI transaction and delivers the remaining data using a memory write command.

The 21555 continues the MWI transaction as long as a full cache line is posted in the posted write queue. A a full cache line corresponds to the cache line size of the target bus. When the 21555 is within one data phase of delivering a complete cache line and there is not another full cache line posted in the queues, the 21555 master terminates the transaction at the cache line boundary. For example: 1 Dword for 32-bit transactions or 2 Dwords for 64-bit transactions. This can occur because:

- The transaction has terminated on the initiator bus at a non-cache line boundary.
- The write data is being pulled from the queue faster than it is being posted. In this, a full cache line is not posted soon enough to continue the MWI.

When the 21555 terminates an MWI transaction before all write data is delivered, it initiates another write transaction to finish delivery of the write data. When a fraction of a cache line remains, the 21555 initiates the transaction with the memory write command. When at least a complete cache line was subsequently posted, then the 21555 once again initiates the transaction with an MWI command.

5.2.3 64-bit Extension Posted Write Transaction

The 21555 uses the 64-bit extension signals, when implemented, for accepting and delivering posted write data.

As a target, the 21555 asserts **ACK64#** in response to the initiator's assertion of **REQ64#** for memory writes and MWI commands if the address is Quadword aligned (address bit **AD[2]** is zero). The 21555 then accepts 64 bits of data per data phase without inserting target wait states.

As an initiator, the 21555 asserts **REQ64#** when delivering posted write data as long as the burst consists of a minimum of 4 Dwords, and the original address is Quadword (64-bit) aligned. When the target asserts **ACK64#,** write data is delivered 64 bits per data phase without inserting master wait states. When the burst ends on an odd Dword address boundary, the 21555 forces the high four byte enables of the last data phase in the burst to be deasserted.

5.2.4 Write Performance Tuning Options

The 21555 implements several features and options that affect write performance when forwarding posted write transactions

5.2.4.1 Memory Write and Invalidate

When the MWI Enable bit in configuration space is set for that corresponding interface, the 21555 is enabled to initiate MWI transactions as described in Section 5.2.2.

5.2.4.2 Fast Back-to-Back

The 21555 may be enabled to initiate fast back-to-back transactions. The 21555 must have the bus grant the clock cycle before it asserts **FRAME#** for the second transaction, and the Fast Back-to-Back Enable bit must be set for the interface on which the 21555 is initiating the transaction. When both of these conditions exist, the 21555 may initiate the second transaction with fast back-to-back timing following a write transaction that is not terminated with **STOP#**.

5.2.4.3 Write-Through

When the 21555 is able to obtain access to the target bus and start transferring write data to the target before the transaction has been terminated on the initiator bus, it automatically enters flow-through mode. In flow-through mode, the 21555 can sustain long write bursts as long as a queue-empty condition is detected in posted write buffers or until an aligned disconnect boundary is reached. A queue-empty condition exists when the number of Dwords left in the posted write buffer is less than an unaligned cache line amount. When the queue-empty condition is detected, the 21555 master terminates the transaction on the target bus. When an aligned disconnect boundary is reached, the 21555 returns a target disconnect on the initiator bus. Flow-through mode behavior is used for both memory write and MWI commands.

5.2.4.4 Memory Write Disconnect Mode

The 21555 implements a Memory Write Disconnect Mode bit in device specific configuration space. When enabled, the 21555 disconnects memory writes on aligned cache line boundaries, using the cache line size corresponding to the target bus

5.2.4.5 Posted Write Queue Tuning

The 21555 implements a posted write queue management control bit for each posted write queue in the Chip Control 1 configuration register. This bit specifies at what threshold the 21555 returns a target retry instead of accepting write data. Setting this bit can minimize fragmentation of posted write transactions and can prevent bursts from being broken into sub-cache line bursts. The tuning options are as follows:

- Target retry is returned when less then a cache line is free.

  For a posted write starting on an odd Dword address, the threshold is CLS-1 Dword entries free.
- Target retry is returned when less then half a cache line is free, for CLS = 8, 16, or 32 Dwords.

When the posted write queue is designated as full, the 21555 returns a target retry to the initiator and does not post any write data. The 21555 uses the Cache Line Size corresponding to the target bus for the target retry threshold.

5.3 Delayed Write Transactions

The 21555 uses delayed transactions when forwarding I/O writes from one PCI interface to the other. Delayed transactions are also used for CSR or configuration register writes that cause the 21555 to initiate a transaction on the opposite interface, such as:

- CSR or configuration register write access that causes the 21555 to initiate a configuration write transaction.
- CSR write access that causes the 21555 to initiate an I/O write transaction.

When an I/O write intended for the opposite PCI bus is first initiated, the 21555 returns a target retry. When the delayed transaction queue is not full and if a transaction having the same address and bus command does not already exist in the delayed transaction queue, the 21555 queues the transaction information:

- Including address.
- Bus command.
- Write data.
- Byte enables.

  Note: The byte enables are not checked when the 21555 decides whether to queue a delayed write transaction.

- When the transaction queued is a result of an I/O Configuration Data register write, the 21555 queues the appropriate data based on the type of access desired, the address and data contained in the corresponding registers, and the byte enables used for the register access. This phase of the delayed transaction is called a delayed write request (DWR).
- The 21555 requests the target bus and initiates the delayed write transaction as soon as the 21555 ordering rules allow. (See Section 5.7). The 21555 always performs a single 32-bit data phase when initiating a delayed write transaction. The 21555 completes the transaction on the target bus and adds the completion status to the queue. Completion status contains the type of termination (**TRDY#**, target abort, master abort) and whether **PERR#** assertion was detected. This phase of the delayed transaction is called the delayed write completion (DWC).

When the 21555 receives 2^{24} consecutive target retries from the target, it discards the delayed write request and conditionally asserts **SERR#** on the initiator bus. See Chapter 12. This retry counter may be disabled by setting the retry counter disable bit in the Chip Control 0 Configuration register. When the transaction is discarded before completion, the 21555 returns a target abort to the initiator.

When the initiator repeats the transaction using the same address, bus command, write data, and byte enables, then the 21555 returns the appropriate target termination when ordering rules allow. Otherwise, the 21555 continues to return target retry. The target terminations are listed in Table 13.

Table 13. Delayed Write Transaction Target Termination Returns

Target Bus Response	Initiator Bus Response
TRDY#	**TRDY**# and **STOP#** when multiple data phases are requested.
Target abort	Target abort
Master abort	• **TRDY#** when Master Abort Mode bit = 0 • Target abort when Master Abort Mode bit = 1

When the 21555 has a delayed completion to return to an initiator, and the initiator does not repeat the transaction before the Master Time-Out Counter for that interface expires, it discards the delayed completion transaction. When enabled to do so, the 21555 asserts **SERR#** on the initiator bus. The Master Time-Out Counter expiration value is either 2^{10} or 2^{15} PCI clock cycles, programmable in the Chip Control 0 configuration register. The Master Time-Out Counter is disabled when the Master Time-Out Disable bit in the Chip Control 0 configuration register is zero.

5.4 Delayed Read Transactions

This section discusses the these Delayed Read Transactions:

- Section 5.4.1, “Nonprefetchable Reads” on page 56.
- Section 5.4.2, “Prefetchable Reads” on page 57.
- Section 5.4.3, “Prefetchable Read Transactions Using the 64-bit Extension” on page 57.
- Section 5.4.4, “Read Performance Features and Tuning Options” on page 57.

The 21555 uses delayed transactions when forwarding any type of read from one PCI interface to the other. Read types are I/O, memory, memory read line, and memory read multiple.

Delayed transactions are also used for parallel ROM reads and CSR or configuration register reads that cause the 21555 to initiate a PCI read transaction, such as:

- CSR or configuration register read access that causes the 21555 to initiate a configuration read transaction.
- CSR read access that causes the 21555 to initiate an I/O read transaction.

The delayed read transaction protocol is similar to that of delayed write transactions, with the exception that 64-bit transfers may be used for delayed-memory read transactions. When an I/O or memory read intended for the other PCI bus is first initiated, the 21555 returns a target retry. The 21555 queues the transaction information if the delayed transaction queue is not full and a transaction having the same address and bus command does not already exist in the delayed transaction queue. This includes address, bus command, byte enables for nonprefetchable reads.

> ***Note:*** The byte enables are not checked when the 21555 decides whether to queue a delayed write transaction.

When the transaction queued is a result of a CSR or configuration register read, the 21555 queues the appropriate data based on the type of access desired, the address contained in the register, and the byte enables used for the data register access. This phase of the delayed transaction is called a Delayed Read Request (DRR).

The 21555 requests the target bus and initiates the delayed read transaction as soon as the 21555 ordering rules allow. See Section 5.7. When the transaction is a nonprefetchable read as described in Section 5.4.1, the 21555 requests only a single Dword of data. When the transaction is a memory read, the 21555 follows the prefetch rules outlined in Section 5.4.2. The 21555 completes the transaction on the target bus and adds the read data and parity to the read data queue and the completion status to the delayed transaction queue. This phase of the delayed transaction is called the Delayed Read Completion (DRC). When the 21555 receives 2^{24} consecutive target retries from the target, the 21555 discards the delayed read transaction and conditionally asserts **SERR#** on the initiator bus. See Chapter 12. This retry counter may be disabled by setting the Retry Counter Disable bit in the Chip Control 0 Configuration register. If the transaction is discarded before completion, the 21555 returns a target abort to the initiator.

When the initiator repeats the transaction using the same address, bus command, and byte enables, then the 21555 returns the read data, parity, and appropriate target termination when ordering rules allow. For all memory read type transactions, the 21555 aliases the memory read, memory read line, and memory read multiple commands when comparing a transaction in the delayed transaction queue to one initiated on the PCI bus. Regardless of the exact command used, when the address matches and both commands are any type of memory read, the 21555 considers it a match. When there is no match, the 21555 is discarding data. When the ordering rules prevent returning the completion at that point, the 21555 returns target retry. The target terminations are listed in Table 14.

Table 14. Delayed Read Transaction Target Termination Returns

Target Bus Response	Initiator Bus Response
TRDY#	**TRDY#** and **STOP#** when returning last data and FRAME# is asserted
Target abort	Target abort
Master abort	**TRDY#** and FFFFFFFFh when Master Abort Mode bit = 0. Target abort when Master Abort Mode bit = 1.

When the 21555 has a delayed completion to return to an initiator, and the initiator does not repeat the transaction before the Master Time-out Counter for that interface expires, then the 21555 discards the delayed completion transaction. When enabled to do so, the 21555 asserts **SERR#** on the initiator bus. The Master Time-out Counter expiration value is either 2^{10} or 2^{15} PCI clock cycles, programmable in the Chip Control 0 configuration register. The Master Time-out Counter is disabled when the Master Time-out Disable bit in the Chip Control 0 configuration register is zero.

5.4.1 Nonprefetchable Reads

The following transactions are considered by the 21555 to be nonprefetchable:

- I/O transactions.
- Configuration transactions.
- Transactions using the memory read command that address a range configured as nonprefetchable.
- Primary bus memory reads to the Expansion ROM BAR.

When initiating a nonprefetchable read, the 21555 requests only a single Dword of read data from the target. The 21555 uses the same byte enables driven by the initiator of the transaction.

When the 21555 returns the read data to the initiator, it asserts **STOP#** with **TRDY#** when the initiator is requesting multiple Dwords.

5.4.2 Prefetchable Reads

The following transactions are considered by the 21555 to be prefetchable read transactions:

- Transactions using the memory read line command.
- Transactions using the memory read multiple command.
- Transactions using the memory read command that address a range configured as prefetchable.

During a prefetchable read, the 21555 speculatively reads data from the target before the initiator explicitly requests it. The amount of data read depends on the read command, the cache line size corresponding to the initiator bus, and whether the 21555 is in flow-through mode, as described in Table 15. The 21555 drives the byte enables to 0h for all data phases, regardless of the byte enables driven by the initiator of the transaction.

When the 21555 returns prefetchable read data to the initiator, it continues to return read data until the master deasserts **FRAME#** and **IRDY#** ending the transaction, or until the 21555 runs out of read data and the target disconnect is returned. When the master terminates the transaction, the 21555 discards the unconsumed read prefetch data. Read data is discarded at a rate of 8 Dwords per clock cycle. During read data discard, the 21555 is unable to return any other delayed transaction completions on the initiator bus or enqueue new delayed requests.

5.4.3 Prefetchable Read Transactions Using the 64-bit Extension

The 21555 uses the 64-bit extension signals when implemented, to initiate and complete prefetchable read transactions.

As a target, the 21555 asserts **ACK64#** in response to the initiator's assertion of **REQ64#** for prefetchable memory read transactions where the 21555 has more than 1 Dword of data to return. The 21555 returns 64 bits of data per data phase without inserting target wait states, with the exception of a temporary queue-empty condition during flow-through. When the 21555 has an odd number of Dwords to return to the initiator, it disconnects before delivering the last Dword. The last Dword is discarded.

As an initiator, the 21555 asserts **REQ64#** for all prefetchable memory reads that have a starting address on an aligned Quadword boundary (that is, address bit **AD[2]** = 0). This prevents the 21555 from accidentally prefetching over an aligned prefetch address boundary. The 21555 then accepts 64 bits of read data per data phase without inserting master wait states.

5.4.4 Read Performance Features and Tuning Options

The 21555 implements several features and options that affect read performance when forwarding prefetchable read transactions.

5.4.4.1 Read Flow-Through

When the bandwidth of the initiator PCI interface is less than or equal to the bandwidth of the target PCI interface, the 21555 may use flow-through, or streaming, operation when returning read data. When the initiator of a delayed prefetchable read transaction repeats the transaction, and the 21555 starts delivering read data on the initiator bus while it is still accepting data for that transaction on the target bus, the 21555 enters read flow-through mode. When in flow-through mode, the 21555 can sustain long read bursts up to a 4KB aligned address boundary, or up to a page address boundary for upstream transactions falling into Memory Range 2. The 21555 always stops the prefetching of reads at 4KB address boundaries. When the read data queue empties while the 21555 is in flow-through mode, the 21555 waits up to seven cycles and then disconnects if read data is still not available.

When using the Quadword boundary, **REQ64#** asserts every time the transaction is Quadword-aligned (**AD[3:0]** = x000b). In some cases, the address is only 2 Dwords away from a cache line boundary, or a 4KB boundary. This means that if an **ACK64#** is not received from the target, another transaction may be necessary to get the high Dword (since **FRAME#** is only asserted for one cycle, indicating a single data phase).

However, when **REQ64#** is only asserted when the address is octaword-aligned, instances occur where **REQ64#** does not assert, but the address is several Dwords from a disconnect or prefetch boundary. For example, when the cache line size is 16 Dwords (32 bytes) but the address is Quadword-aligned on Dword address 2 (xxxx xx08h), there are still 14 Dwords to deliver before the end of a cache line is reached.

Assuming an even distribution of unaligned addresses (which are a minority of all transactions), it is more efficient to optimize for the 64-bit behavior and assert **REQ64#** on Quadword-aligned accesses, while losing some efficiency on those Quadword-aligned transactions near a boundary where **ACK64#** is not asserted.

When the bandwidth of the initiator interface is twice that of the target interface, then limited flow-through is allowed. For example, when the initiator is performing a 64-bit transaction, but the corresponding transaction to the target is a 32-bit transaction. In this case, for cache line sizes of 16 and 32 Dwords, flow-through is allowed only when a full cache line is accumulated in the read data buffer for memory read multiple transactions, or when half a cache line is accumulated for memory read line and prefetchable memory read transactions. For cache line sizes of 8 Dwords, flow-through is allowed when 2 Dwords have been accumulated. In limited flow-through only the standard prefetch boundaries described in **Prefetching** are used, that is, longer bursts are not accommodated. However, limited flow-through minimizes the latency when returning read data with a 2:1 bandwidth mismatch. When the read data queue empties while the 21555 is in limited flow-through mode, the 21555 waits up to seven cycles and then disconnects if read data is still not available.

When the bandwidth of an initiator interface is four times that of the target (e.g., the initiator is performing a 66 MHz, 64-bit operation), and the target is operating at 33 MHz, 32-bit operation, no flow-through is performed. The read operation must complete at the target before read data is returned to the initiator. This is to prevent inserting wait states and possible early disconnects on the initiator bus.

5.4.4.2 Prefetching

The 21555 prefetches read data to the aligned address boundaries listed in Table 15.

Table 15. Prefetch Boundaries

Read Command	Non-prefetchable Range	Prefetchable Range	In Flow-Through Mode
Memory Read	1 Dword	1 cache line	Page boundary for transactions in Upstream Memory 2 range 4KB boundary Initiator deasserts **FRAME#**
Memory Read Line	1 cache line	1 cache line	Page boundary for transactions in Upstream Memory 2 range 4 KB boundary Initiator deasserts **FRAME#**
Memory Read Multiple	2 cache lines	2 cache lines	Page boundary for transactions in Upstream Memory 2 range 4 KB boundary Initiator deasserts **FRAME#**

The cache line size corresponding to the initiator bus is used for determining prefetch boundaries.

5.4.4.3 Read Queue Full Threshold Tuning

The 21555 implements read queue management control bits for each read data queue in the Chip Control 1 configuration register. These bits specify at what read-queue threshold the 21555 initiates a delayed prefetchable read transaction on the target bus. Use of these bits can minimize fragmentation of prefetchable read bursts. The encoding and behavior of these bits are as follows:

- 00b: at least eight Dwords free in read data queue for all memory read commands.
- 01b: at least eight Dwords free for all memory read commands (same as 00b).
- 10b: at least one cache line free for MRL and MRM, eight Dwords free for memory read.
- 11b: at least one cache line free for all memory read commands .

In these cases, the initiator bus cache line size is used. When the cache line size is not set to a valid value, 8 Dwords is used for the read queue threshold.

For non-prefetchable memory reads, a threshold of 8 Dwords (one read queue block) is always used.

5.5 64-Bit and 32-Bit Transactions Initiated by the 21555

The 21555 requests a 64-bit transaction on the primary or secondary bus 64-bit PCI extension by asserting **p_req64_l** on the primary bus or **s_req64_l** on the secondary bus, respectively, during the address phase.

The 21555 asserts and deasserts **REQ64#** during the same cycles in which it asserts and deasserts **FRAME#**, respectively.

Under the following specific circumstances, the 21555 does not use the 64-bit extension when initiating transactions and therefore does not assert **REQ64#**:

- Signal **p_req64_l** was not asserted by the primary bus central function during reset for upstream transactions only. The 64-bit extension is not supported on primary PCI bus.
- The 21555 is initiating an I/O transaction.
- The 21555 is initiating a configuration transaction.
- The 21555 is initiating a nonprefetchable memory read transaction.
- The 21555 is initiating a special cycle transaction.
- The address is not Quadword aligned (**AD[2]** = 1).
- 3 Dwords or less in posted right buffer.
- The 21555 is resuming a memory write transaction after a target disconnect, and **ACK64#** was not asserted by the target in the previous transaction. (This does not apply when the previous target termination was a target retry.)
- A single Dword read transaction is being performed.
- The address is near the top of a cache line (**AD[3]** = 1) applies to prefetchable read transactions.

5.6 Target Terminations

This section describes the following target retries, target disconnects, and target aborts received and returned by the 21555.

5.6.1 Target Terminations Returned by the 21555

The 21555 returns a target retry under the following circumstances:

- Queue is full for posted memory writes.
- Delayed transaction is queued but response is not ready.
- Queue is full for delayed transactions. The delayed transaction is not queued.
- Serial preload is ongoing.
- Primary Lockout Bit is set for primary bus transactions.
- Transaction is in progress for CSR generation of I/O or Configuration Access (delayed transaction not ready).
- The 21555 is discarding read data.

Target disconnects by the 21555 always consist of **STOP#** asserted and **TRDY#** deasserted (that is, a target disconnect without data transfer). The 21555 returns a target disconnect under the following circumstances:

- Queue fills during posted write.
- Cache line boundary is reached for MWI transaction and the 21555 cannot buffer another cache line.
- Cache line boundary is reached for memory write transaction and the Memory Write Disconnect bit is set.
- The 21555 runs out of read data during completion of delayed transaction to the initiator.
- The 21555 is responding to a nonprefetchable Read transaction if multiple data phases are requested by the initiator.
- Multiple data phases requested by the initiator for an I/O or configuration access.
- Low two address bits of the transaction are non-zero.

The 21555 returns a target abort and sets the Signaled Target Abort bit in the Primary and Secondary Status register under the following circumstances:

- Target abort is detected during a delayed transaction completion on the target bus.
- Master abort is detected in response to a delayed transaction on the target bus when the Master Abort Mode bit is set to a 1. See Table 77, “Chip Control 0 Register” on page 156.
- Delayed transaction request is discarded after 2^{24} target retries received from the target.
- Invalid lookup table entry is encountered when forwarding upstream transactions in Upstream Memory 2 range and the Master Abort Mode bit is set to a 1.

5.6.2 Transaction Termination Errors on the Target Bus

When the 21555 detects a target abort on the target bus, the 21555 sets the Received Target Abort in the Primary and Secondary Status register. See Table 62, “Primary and Secondary Status Registers” on page 150. In addition, the 21555:

- For delayed transactions, returns a target abort to the initiator and sets the Signaled Target Abort bit in the Primary and Secondary Status register.
- For posted write transactions, asserts **SERR#** on the initiator bus if the **SERR#** Enable for that interface is set, and sets the Signaled System Error bit in the Primary and Secondary Status register.

When the 21555 detects a master abort on the target bus, the 21555 always sets Received Master Abort bit in Primary and Secondary Status register. In addition, the 21555:

- For delayed transactions when the Master Abort Mode bit is 0, returns **TRDY#** and, for reads, FFFFFFFFh to the initiator. See Table 77, “Chip Control 0 Register” on page 156.
- For delayed transactions when the Master Abort Mode bit is 1, returns a target abort and sets the Signaled Target Abort bit in the Primary and Secondary Status register.
- For posted write transactions, assert **SERR#** and set the Signaled System Error bit on the initiator bus if the **SERR#** Enable for that interface is set and the **SERR#** Disable for Master Abort during Posted Write is clear.

5.7 Ordering Rules

The 21555 can queue and forward multiple transactions at once. Therefore, at any one time the 21555 has multiple posted write and multiple delayed transaction requests and completions queued and traveling in the same and opposite directions. The 21555 uses a set of ordering rules to dictate the order in which it initiates posted writes, initiates delayed transaction requests, and returns delayed transaction completion status. These rules reflect both the ordering constraints outlined in the *PCI Local Bus Specification*, *Revision 2.2* as well as implementation choices specific to the 21555.

Independent transactions on the primary and secondary buses only have a relationship when those transactions cross the 21555. General ordering guidelines for transactions crossing the 21555 are:

- The ordering relationship of a transaction with respect to other transactions is determined when the transaction completes; that is, when a transaction ends with a termination other than target retry.
- Requests terminated with target retry may be accepted and completed in any order with respect to other transactions that have been terminated with target retry. When the order of completion of delayed requests is important, the initiator should not start a second delayed transaction until the first one has been completed. When more than one delayed transaction is initiated, the initiator should repeat all the delayed transaction requests using some fairness algorithm; that is, reattempting a delayed transaction cannot be contingent on completion of another delayed transaction, otherwise a deadlock can occur. This deadlock is avoided with an out-of-order delivery and completion.
- Write transactions flowing in one direction have no ordering requirements with respect to write transactions flowing in the other direction. The 21555 can accept posted writes on both interfaces at the same time, as well as initiate posted writes on both interfaces at the same time.
- The acceptance of a posted memory write as a target can never be contingent on the completion of a non-posted transaction as a master. This is true of the 21555 and must also be true of other bus agents; otherwise, a deadlock can occur.
- The 21555 accepts posted writes regardless of the state of completion of any delayed transactions being forwarded across the bridge.

- A target retry in response to a posted write is allowed, but only due to temporary conditions, such as a buffer-full condition.

The ordering rules apply to transactions crossing the bridge in the same direction.

— A posted write.

— A delayed write and read request.

— A delayed write and read completion.

Delayed completions cross the bridge in the opposite direction of its respective delayed request. Table 16 lists the 21555 transaction ordering rules.

Table 16. 21555 Transaction Ordering Rules

↓ Pass→	Posted Write	Delayed Read Request	Delayed Write Request	Delayed Read Completion	Delayed Write Completion
Posted Write	No	Yes	Yes	Yes	Yes
Delayed Read Request	No	Yes/No†	Yes/No†	Yes	Yes
Delayed Write Request	No	Yes/No†	Yes/No†	Yes	Yes
Delayed Read Completion	No	Yes	Yes	Yes	Yes
Delayed Write Completion	Yes	Yes	Yes	Yes	Yes

† Dependent on the state of the Delayed Transaction Ordering bit.

The only ordering restriction the 21555 enforces is ordering with respect to posted writes. No other transaction other than a delayed write completion can pass a posted write. Posted writes are delivered in the order in which they are accepted.

Delayed transactions may be initiated by the 21555 in any order, and are not necessarily initiated in the order in which they are received. When the 21555 initiates a delayed transaction, the 21555 can do the following:

- When the Delayed Transaction Order Control configuration bit is not set, the 21555 uses a rotating fairness algorithm to select which delayed transaction it initiates next, regardless of the type of target termination is returned (retry, **TRDY#**, etc.).
- When the Delayed Transaction Order Control configuration bit is set, the 21555 continues to initiate the same transaction until a response other than target retry is received.

Note: Performance may be affected if the Delayed Transaction Order Control bit is set, as the 21555 deasserts the PCI request signal between transactions. When the Delayed Transaction Order Control bit is zero, the 21555 may keep **REQ#** asserted after a target retry or target disconnect if another transaction is pending. See Table 77, “Chip Control 0 Register” on page 156.

Delayed completions are returned to the initiator when ready, regardless of the order in which corresponding delayed requests were queued. A delayed read completion may not be returned to the initiator (the initiator receives a target retry) when a posted write is ahead of the delayed completion in the queues. That is, the write was posted in the direction of the completion, but before the read data was queued. In this case, the write must be delivered before the read data can be returned to the initiator.

Initialization Requirements 6

This chapter presents the theory of operation information about the 21555 initialization requirements. See Chapter 16 for specific information about the initialization registers.

6.1 Power Management, Hot-Swap, and Reset Signals

Table 17 describes the power management, hot-swap, and reset signals.

Table 17. Power Management, Hot-Swap, and Reset Signals (Sheet 1 of 2)

Signal Name	Type	Description
l_stat	TS	CompactPCI hot-swap local status pin. As an input to the 21555, this signal indicates the sense of the ejector switch and therefore the state of the LED in a CompactPCI card supporting distributed hot-swap. As an output from the 21555, it controls the LED. When CompactPCI hot-swap is not supported by the add-in card, this signal should be tied low with a 1k resistor.
p_enum_l	OD	Primary bus CompactPCI hot-swap event. Conditionally asserted by the 21555, this signal indicates either that the card has been inserted and is ready for configuration, or that the card is about to be removed. This signal is deasserted when the corresponding insertion or removal event bit is cleared. This signal should be pulled up by an external resistor.
p_pme_l	OD	Primary bus power management event. Provides power management signaling capability on behalf of the subsystem. The 21555 asserts **p_pme_l** when all of the following are true: • Signal **s_pme_l** is asserted low. • Signal **p_pme_l** is supported in the current power state. • **PME_EN** bit is set. (See Table 120, “Power Management Control and Status Register” on page 187.) Once asserted, **p_pme_l** is deasserted when the PME status bit or the **PME_EN** bit is cleared. If the **PME#** isolation circuitry is needed, it must be implemented externally.
p_rst_l	I	Primary PCI bus **RST#**. Signal **p_rst_l** forces the 21555 to a known state. All register state is cleared, and all PCI bus outputs are tristated, with the exception of **s_ad**, **s_cbe_l**, and **s_par** if the 21555 is designated as the central function. Tristated signals are: • **p_perr_l** • **p_serr_l** • **p_inta_l** • **p_enum_l** • **p_pme_l** • **p_req_l** • **s_perr_l** • **s_serr_l** • **s_inta_l** • **s_gnt_l [8:0]**. Signal **p_rst_l** is asynchronous to **p_clk**.

Table 17. Power Management, Hot-Swap, and Reset Signals (Sheet 2 of 2)

Signal Name	Type	Description
s_pme_l	I	Secondary bus power management event. The subsystem asserts this signal to the 21555 to indicate that it is signaling a power management event. The 21555 conditionally asserts **p_pme_l** when **s_pme_l** is asserted low. When the subsystem does not generate power management events, this signal can also be used for a subsystem status signal. A deasserting (rising) edge on this signal can conditionally cause the 21555 to assert **p_inta_l**. When this signal is not used, it should be tied high with a 1k resistor.
s_rst_in_l	I	Alternate reset input for the 21555. Asserting **s_rst_in_l** is the same as asserting **p_rst_l**. These two signals are ORed on the 21555. All configuration modes are captured on this edge. Signal **s_rst_in_l** allows for either a reset to be initiated from the secondary bus or a board reset for a hot-swap.
s_rst_l	O	Secondary PCI bus **RST#**. Signal **s_rst_l** is driven by the 21555 and acts as the PCI reset for the secondary bus. The 21555 asserts **s_rst_l** when *any* of the following conditions are met: • Signal **p_rst_l** is asserted. • The secondary reset bit in the Table 123, “Reset Control Register” on page 188 in configuration space is set. • The chip reset bit in the Table 123, “Reset Control Register” on page 188 in configuration space is set. • Power management transition from $D3_{hot}$ to D0 occurs. When the 21555 asserts **s_rst_l**, it tristates all secondary control signals and, when designated as the secondary bus central resource, asserts **s_req64_l** and drives zeros on **s_ad**, **s_cbe_l**, and **s_par**. Signal **s_rst_l** remains asserted until **p_rst_l** is deasserted, and the secondary reset bit is clear. Deassertion of **s_rst_l** occurs automatically based on internal timers when **s_rst_l** assertion is caused by setting the chip reset bit or a power management transition. Assertion of **s_rst_l** by itself does not clear register state, and configuration registers are still accessible from the primary PCI interface.

6.2 Reset Behavior

The 21555 implements a primary reset input, **p_rst_l**, a secondary reset input **s_rst_in_l**, and a secondary reset output, **s_rst_l**. The 21555 also implements a Chip Reset bit and a Secondary Reset bit in the Table 123, “Reset Control Register” on page 188.

The device is reset when one of the following occurs:

- The signal **p_rst_l** is asserted.
- The signal **s_rst_in_l** is asserted.
- The Chip Reset bit is written with a 1.
- A power management transition from $D3_{hot}$ to D0 occurs (see Section 6.4.1).

When the Chip Reset bit is written with a 1, the chip reset bit is cleared 7 clocks after it is set. The actual chip reset signal is internally delayed to allow the configuration cycle to complete normally. Chip reset causes all the register values to be reset, and all the queues to be cleared. The primary PCI bus and control signals are tristated as long as either chip reset is occurring or **p_rst_l** is asserted.

The secondary reset output, **s_rst_l**, is asserted and remains asserted when any of the following are true:

- The 21555 primary reset input, **p_rst_l**, is asserted.
- The 21555 secondary reset input, **s_rst_in_l**, is asserted.
- The Secondary Reset bit in the Table 123, “Reset Control Register” on page 188 is set to a 1.
- The Chip Reset bit in the Table 123, “Reset Control Register” on page 188 is set to a 1.
- A power management transition from $D3_{hot}$ to D0 occurs (see Section 6.4.1).

A power management transition from $D3_{hot}$ to D0 or setting the Chip Reset bit causes the Secondary Reset bit to set automatically. When set automatically, the Secondary Reset bit also clears automatically and **s_rst_l** deasserts after greater than 100 µs following **s_rst_l** assertion.

Assertion of **s_rst_l** by setting the secondary reset bit does not cause the 21555 register state to be reset. However, all the 21555 data buffers are reset.

Note: A configuration write is required to clear the secondary reset bit if the bit is set by a configuration write. Care must be taken when this bit is asserted from the secondary interface.

Table 18 summarizes the various 21555 reset mechanisms.

Note: The signal **s_rst_l** is asserted for all reset mechanisms, but how **s_rst_l** deasserts and whether the device is reset varies from case to case.

Table 18. Reset Mechanisms

Reset Mechanism	Reset 21555 Buffers and State	Assert Secondary Reset Bit	Deassertion of s_rst_l
p_rst_l	Yes	No	On **p_rst_l** deassertion
s_rst_in_l	Yes	No	On **s_rst_in_l** deassertion
Chip Reset Bit set	Yes	Yes	Automatically after >100 ms (Secondary Reset bit also clears automatically)
Secondary Reset Bit set	Reset data buffers and primary master state machine	Yes	On clearing of Secondary Reset bit
Transition from $D3_{hot}$ to D0 (see Section 6.4.1).	Yes	Yes	Automatically after >100 ms (Secondary Reset bit also clears automatically)

6.2.1 Central Function During Reset

The 21555 is selected to be the secondary bus central function when it detects **pr_ad[6]** low when **s_rst_l** is asserted. When the 21555 detects this condition, it immediately drives **s_ad, s_cbe_l**, and **s_par** low and tristates secondary bus control signals for the duration of secondary bus reset. When the 21555 implements a 64-bit secondary interface, it also asserts **s_req64_**l, but tristates all other secondary bus 64-bit extension signals.

When **pr_ad[6]** is detected high during **s_rst_l** assertion, another device is acting as a central function on the secondary bus. The 21555 tristates all secondary PCI signals, including **s_ad**, **s_cbe_l**, and **s_par** for the duration of secondary bus reset. The 21555 does not assert **s_req64_l** during reset and an external agent must assert **s_req64_l** to enable the 21555's secondary interface 64-bit extension.

> ***Note:*** The signal **s_rst_in_l** assertion causes **s_rst_l** to asynchronously assert. When secondary bus central functions are enabled, these functions continue to activate upon assertion of **s_rst_l**.

6.3 21555 Initialization

The 21555 supports the following mechanisms for initialization and configuration:

- Preconfiguration using the SROM interface, this is also called SROM preload.
- Configuration by the local processor through the secondary interface.
- Configuration by the host processor through the primary interface.

Initialization may use all of these mechanisms, or only a subset. Initialization must take place:

- After a hardware reset caused by **p_rst_l** or **s_rst_in_l** assertion.
- After device reset caused by setting the Chip Reset bit in the Table 123, "Reset Control Register" on page 188.
- After device reset caused by a power management transition from $D3_{hot}$ to D0.

The 21555 reset consists of the following sequence:

1. Signal **p_rst_l** or **s_rst_in_l** asserts or the device is reset. The 21555 tristates all PCI outputs and asserts **s_rst_l**.
2. Signal **p_clk** and **s_clk** start; **s_clk_o** is a buffered version of **p_clk**.
3. When **pr_ad[6]** is low, the 21555 drives **s_ad**, **s_par**, and **s_cbe_l** low for the remainder of **s_rst_l** assertion, and asserts **s_req64_l**.
4. Upon deassertion of **p_rst_l** or **s_rst_in_l**, or on the 1st clock cycle following the completion of chip reset:
   - The value of **pr_ad[3]** specifies the value of the Primary Lockout Reset Value configuration bit upon completion of reset.
   - When **pr_ad[4]** is low, the 21555 switches into synchronous mode.
   - When **pr_ad[5]** is low, **s_clk_o** is disabled and driven low.
   - When **pr_ad[7]** is low, the internal arbiter is disabled.
5. The 21555 deasserts **s_rst_l** after **p_rst_l** or **s_rst_in_l** deassertion, or after 100 µs following **s_rst_l** assertion.

6.3.1 With SROM, Local, and Host Processors

The following is the 21555 initialization procedure using all configuration mechanisms:

1. **Serial Preload**

   Upon deassertion of **p_rst_l** or completion of chip reset, the 21555 automatically starts the serial load sequence when a SROM is present. The serial load takes approximately 18700 primary bus clock (p_clk) cycles (550 SROM clock cycles). During this time, the 21555 returns a target retry to any configuration transaction access from either interface.

   The serial load can overwrite selected PCI read-only registers, program forwarding BAR types and sizes, and configure device-specific configuration registers.

2. **Local Processor Initialization**

   When the serial load is complete, the 21555 configuration registers are now accessible by the local processor from the secondary interface.

   When the Primary Lockout Reset Value bit is set, the 21555 continues to return target retry to any configuration accesses from the primary interface (with the exception of the Reset Control configuration register at offset D8h). The local processor can write selected locations that are loadable from the SROM, and therefore can be used to change parameters loaded during SROM preconfiguration. The local processor can also perform standard PCI configuration of the secondary interface configuration registers. Once the base address registers are mapped and the secondary enables set, the 21555 may accept memory or I/O transactions to its CSR registers.

   When local processor initialization is complete, the local processor should clear the Primary Lockout Reset Value bit to allow host initialization. When the Primary Lockout Reset Value bit is clear after serial preload, the host processor and local processor can access the 21555 concurrently immediately after serial preload is complete. The local processor should *not* change any primary interface preload values that can affect host configuration.

   This mode of initialization is not recommended unless special care is taken that registers are accessed and initialized in their proper sequence.

3. **Host Initialization**

   After the serial preload and Primary Lockout Reset Value bit is clear, the host may perform the standard PCI device configuration. Device-specific expansion ROM code can be accessed through the Primary Expansion ROM Base Address register.

4. **Normal Operation**

6.3.2 Without Serial Preload

A SROM is supported, but not required, for the 21555 preinitialization. In the case where a SROM is not connected to the 21555 or when the first data bits read does not contain 10b, the 21555 terminates the SROM read and configuration space is then available for local processor configuration. The Primary Lockout Reset Value bit can be set to a 1 by pulling **pr_ad[3]** high during chip reset. All primary bus configuration accesses (with the exception of location D8h) then receive target retry until the local processor clears the Primary Lockout Reset Value bit.

The local processor first must preconfigure registers that would have been preloaded by the SROM. This is particularly true of the size and types of the base address registers for forwarding transactions, which upon completion of reset are disabled and request no address space.

Once the local processor preconfigures the necessary registers, normal PCI configuration of the secondary configuration registers can proceed. The local processor then *must* clear the Primary Lockout Reset Value bit to allow access from the primary bus, unless the Primary Lockout Reset Value reset value was designated to be low by pulling **pr_ad[3]** low during reset.

The remainder of the 21555 configuration proceeds as described in Section 6.3.1.

6.3.3 Without Local Processor

Initialization of the 21555 is possible without a local processor, or without local processor intervention. Serial preload is still performed as described in (Section 16.10). However, the serial load *must* clear the Primary Lockout Reset Value bit to allow access of configuration registers from the primary interface. The serial preload must successfully preconfigure the forwarding BAR setup registers, as well as overwrite primary read-only registers as necessary.

Upon completion of the serial preload, all configuration registers are accessible for PCI configuration from the host on the primary bus. The host is then also responsible for configuring the secondary interface and device-specific configuration registers.

6.3.4 Without Local Processor and Serial Preload

When neither the SROM nor a local processor is present, only the reset values of all the read-only registers are used, and all forwarding BARs are disabled and do not request space (since all these registers are set up from the secondary side only). The 21555 configuration registers are accessible, and the 21555 CSR registers can still be mapped into memory or I/O space. However, the Table 107, “Primary Expansion ROM BAR” on page 175 is disabled. A parallel ROM (PROM) can still be accessed through the CSR mechanism. Configuration and I/O transactions can be forwarded through the indirect CSR mechanism; the I20 message unit, doorbell registers, and scratchpad registers are all accessible. The 21555 configuration registers that are accessible only from the secondary interface can be written using the downstream indirect configuration mechanism.

6.3.5 Without Host Processor

Initialization of the 21555 can be performed without a host processor. In this case, the local processor must perform the initialization of the primary configuration registers from the secondary interface.

6.4 Power Management Support

The 21555 implements the PCI Power Management interface on behalf of the subsystem. The 21555 Power Management interface is designed to be flexible to meet the varying needs of different types of subsystem functions. To fully understand the PCI Power Management interface, please refer to the *PCI Power Management Specification, Rev 1.0.* Some functions may need minimal power management support: the D0 and $D3_{hot}$ power states, without **PME#** support. Other functions may need all four power states and **PME#** support. Power management setup is done by SROM preload.

The SROM preload allows the following power management parameters to be defined:

- Power Management revision number.
- D1 power management state support.
- D2 power management state support.
- PME# support.
- Power Management Data register support.
- Device Specific Initialization status bit.

6.4.1 Transitions Between Power Management States

The 21555 is put into a different power state by writing the Power State bits in the Power Management Control and Status configuration register. Table 19 shows the actions that the 21555 takes when transitioning between power states. Although any transition to a lower power state is allowed, all transitions to a higher power state must go to D0.

Table 19. Power Management Actions

Original Power State	Next Power State	Action
D0	D1	No action. Subsystem should have been notified by driver.
D0, D1	D2	No action. Subsystem should have been notified by driver.
D0, D1, D2	$D3_{hot}$	No action. Subsystem should have been notified by driver.
Any State	$D3_{cold}$	No action. Powered off.
D1, D2	D0	Set “Transition to D0” status bit and assert **s_inta_l** when not masked for that event.
$D3_{hot}$	D0	The 21555 performs a chip reset and asserts **s_rst_l** for 100 ms. The 21555 performs a serial preload as soon as chip reset is complete.†
$D3_{cold}$	D0	Power on. Primary bus reset asserts. No special action needed.

† To adhere to the $D3_{hot}$ to D0 recovery time stated in the Power Management Specification, the local processor may have to initialize the 21555 and clear the Primary Lockout Reset Value bit early in the subsystem initialization process.

6.4.2 PME# Support

The 21555 provides optional **PME#** support. Since the 21555 provides the subsystem Power Management Interface registers, the 21555 must also be the source of the **PME#** signal for the subsystem. The 21555 implements a primary bus **PME#** output signal, **p_pme_l**, that is asserted when the subsystem wants to generate a power management event. The 21555 implements a secondary bus power management input signal, **s_pme_l**, that the subsystem asserts to notify the 21555 of this power management event.

The 21555 asserts **p_pme_l** when all of the following are true:

- The 21555 detects **s_pme_l** asserted low.
- **PME#** support for the current power state of the 21555 is enabled, as indicated in the Power Management Capabilities register.
- The PME_En bit is set to a 1 in the Power Management Control and Status register.

When the first two conditions have both been met, the 21555 sets the PME Status bit in the Power Management Control and Status register.

Once **p_pme_l** has been asserted, the 21555 deasserts the signal if either of the following conditions are true:

- The PME Status bit is cleared in the Power Management Control and Status register.
- The PME_En bit is cleared in the Power Management Control and Status register.

The 21555 assumes that **s_pme_l** is deasserted before the PME_Status bit is cleared in the Power Management Control and Status register. Otherwise, multiple assertions of **p_pme_l** may occur. When **PME#** isolation circuitry is required on the primary interface, it must be implemented externally.

6.4.3 Power Management Data Register

The PCI Power Management specification defines an optional data register that can be used for static or dynamic data reporting. A Data Select field in the Power Management Control and Status register selects the type of data to be reported. A Data Scale register provides the scale factor for this data. The 21555 allows implementation of this Data register for static data reporting for the subsystem. The Data Scale value and eight possible data values are loaded into the 21555 through the SROM preload operation. The Power Management Data Enable bit in the serial preload sequence enables the use of these data values; otherwise the Data register reads as 0. The value contained in the Data Select field selects which data byte is to be returned when the Power Management Data register is read.

6.5 CompactPCI Hot-Swap Functionality

The 21555 implements hot-swap functionality that allows it to function as a CompactPCI hot-swap controller. This means that the software connection control interface for CompactPCI hot-swap is implemented. However, bus precharge is not implemented. Please refer to the *CompactPCI Hot-Swap Specification* for more information on CompactPCI hot-swap.

The basic components of a CompactPCI Hot-Swap device are:

- Table 125, "CompactPCI Hot-Swap Control Register" on page 189, at offset ECh.
  - — **ENUM#** Interrupt Mask. Must be clear to enable the assertion of **p_enum_l**.
  - — LOO (LED On/Off), LED software control bit. When set, the 21555 drives **l_stat high**, causing the LED to turn on. When clear, the 21555 drives **l_stat** low, causing the LED to turn off.

    Note: The 21555 periodically tristates **l_stat** for a few cycles to sample the state of the microswitch.
  - — INS_STAT, Insertion status bit.
  - — REM_STAT, removal status bit.
- Support of the hot-swap event pin, **p_enum_l**. This signal is routed to the host CPU through the CompactPCI connector. This signal informs the CPU that the configuration of the system has changed; that is, the card has been inserted or is about to be removed.
- Support bi-directional pin, **l_stat**. This signal functions as both a micro-switch sensor input and a LED control output.

  Note: 2 ms of debounce is implemented on the l_stat pin.

6.5.1 Overview of CompactPCI Controller Hardware Interface

On the connector side, a CompactPCI hot-swap board has a staggered pin arrangement to allow power/ground, signal, and a board inserted indicator to be connected and disconnected in stages. Power and ground are 1st make, last (3rd) break pins. The signal pins are 2nd make, 2nd break pins. The board inserted signal (**BDSEL#**), which is routed to the power conditioning and local reset logic, is last (3rd) make, 1st break.

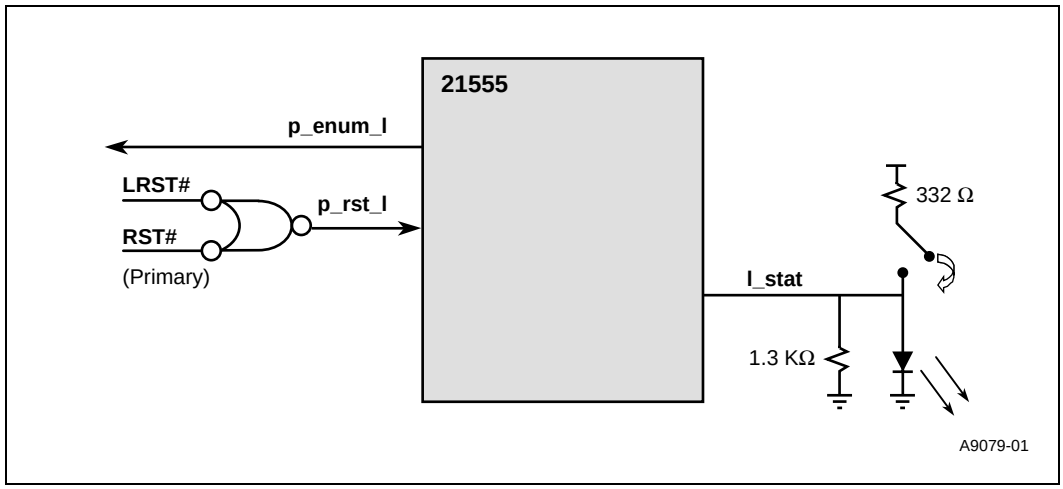
On the board handle side, a card ejector handle controls a micro-switch on the card. When a seated card is removed, the first thing that occurs is that the ejector handle is opened. This causes the micro-switch to close. Similarly, when a card is inserted, the ejector handle is initially open, and then closed when the board is seated. When the ejector handle is closed, the micro-switch on the card opens. The micro-switch state is an input to the CompactPCI hot-swap controller.

A CompactPCI hot-swap card also implements an indicator LED. When the LED is on, this indicates that the board can be removed from the slot. Software may choose to flash the LED to indicate an intermediate state as well. The CompactPCI hot-swap controller controls the state of the LED.

The 21555 multiplexes the microswitch state input and the LED control output onto a single shared pin, **l_stat**. The 21555 both samples this signal to determine the micro-switch state and drives this signal to control the LED. It is assumed that onboard debouncing circuitry is used to ensure that a clean edge is provided for the **l_stat** signal. Figure 11 on page 73 shows how the **l_stat** signal can be used on a CompactPCI hot-swap card. Whenever the 21555 drives **l_stat**, usually when the LOO bit is set, but also in the *Signal Removal* state, it automatically and periodically tristates the **l_stat** signal to sample the state of the micro-switch. Every 1 ms, the 21555 tristates for 8 primary PCI clock cycles to sample its state, when the primary clock is 33 MHz. When the primary clock is faster than 33 MHz (**p_m66ena** is asserted), then the number of cycles for tristated and driving is doubled.

The card's local reset signal, which is asserted upon card removal or insertion, may be OR'ed with the primary bus reset on the card, and then input to the 21555's **p_rst_l** reset input. Alternatively, the secondary reset input **s_rst_in_l** can be used as a local reset input. However, if **s_rst_in_l** is used, **p_req64_l** is not sampled to determine whether to enable the 64-bit extension. Instead, **pr_ad[1]** is sampled and must be pulled up or down to disable or enable the primary bus 64-bit extension.

Figure 11. CompactPCI Hot-Swap Connections



6.5.2 Insertion and Removal Process

Figure 12 is the 21555 Hot-Swap the insertion and removal process. The flow begins from card insertion. This occurs when reset: either **p_rst_l** or **s_rst_in_l**, is asserted and **l_stat** is sampled high.

In the *Local Reset state*, all outputs are tristated, except the secondary reset output, **s_rst_l,** and (conditionally) **s_req64_l** which are driven low. The secondary bus **AD[31:0]** contents are tristated if CFN is not strapped during reset. The state of the micro-switch controls the state of the LED in the Local Reset state. As long as the micro-switch is closed in this state, pulling **l_stat** high, the LED is on. the 21555 does not drive **l_stat** in this state.

When both of the reset input signals are detected high (deasserted), the 21555 enters the *Serial Preload* state. In this state, the 21555 responds to all transactions with target retry. As long as the micro-switch is closed in this state, pulling **l_stat** high, the LED is on. When the micro-switch closes, **l_stat** is pulled low and the LED turns off. When the serial preload completes but the lockout bit is still set, the 21555 remains in the *Serial Preload* state. The Hot-Swap Control register is still not accessible from the primary side, but can be accessed from the secondary side. Therefore, it is possible to control the LOO bit, and force the LED on, from the secondary side.

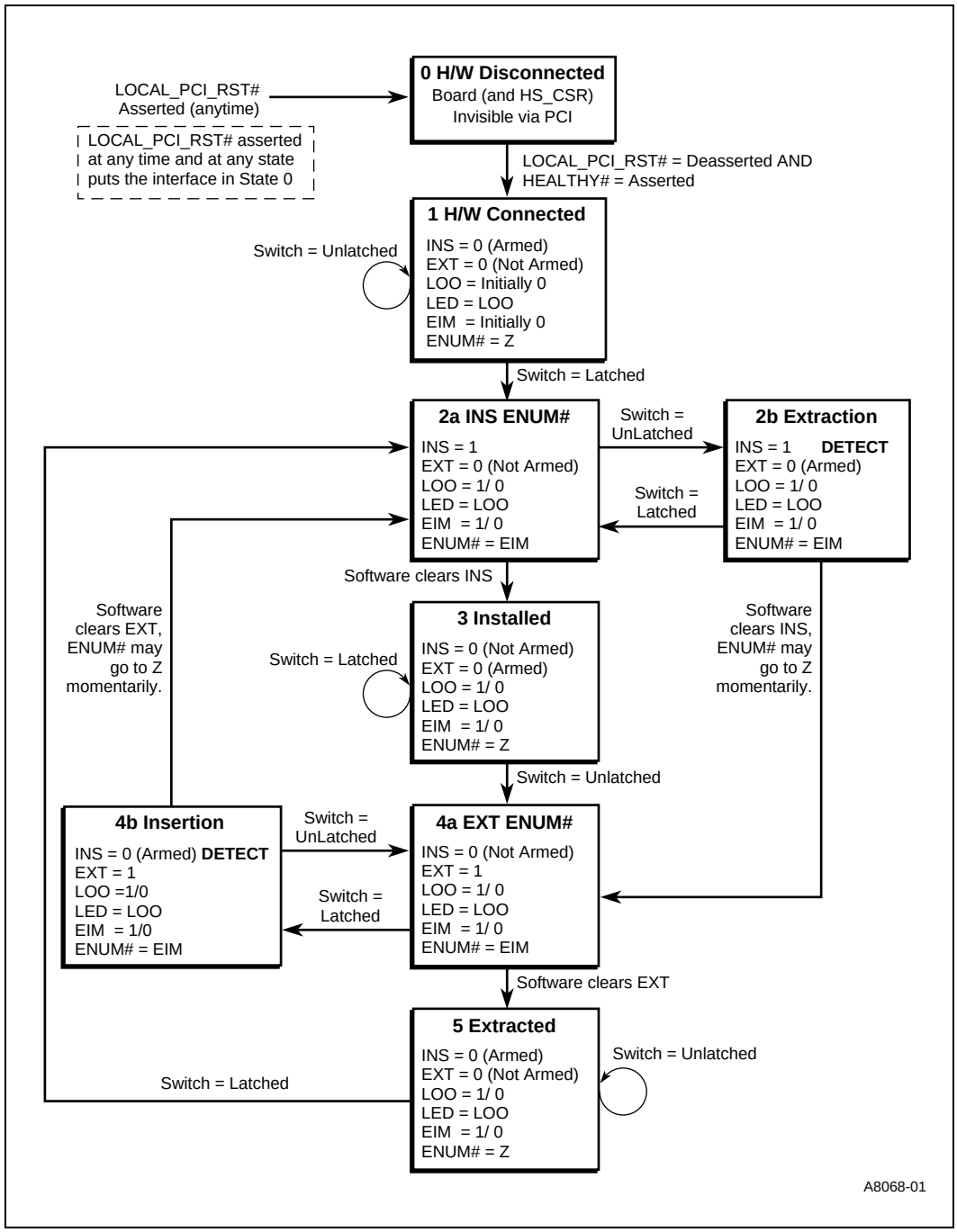
The 21555 enters the *Signal Insertion* state from the *Serial Preload* state when the following conditions are satisfied:

- Serial preload is complete.
- Primary Lockout Reset Value bit cleared.
- Ejector handle is closed (micro-switch opens, and **l_stat** is sampled low).

The card has now been completely seated and the local initialization is complete. The card is ready for host configuration and initialization. The 21555 sets the INS_STAT bit and asserts **p_enum_l** to the host. Upon detecting **p_enum_l** asserted the host processor initializes the card. When the initialization is complete, the host clears the INS_STAT bit.

When the INS_STAT bit is cleared, the card is ready for normal operation. When **l_stat** continues to be sampled low, that indicates that the ejector handle is closed (and the micro-switch is open), meaning the card remains fully inserted. The 21555 enters the *Normal Operation* state.

Figure 12. 21555 Hot-Swap Insertion and Removal



However, when the 21555 samples **l_stat** high once the INS_STAT bit is cleared, this indicates that the ejector handle has been opened. This is interpreted as a removal event, and the 21555 enters the Signal Removal state instead. The same is true when the 21555 samples **l_stat** high while in the Normal Operation state.

When the 21555 enters the Signal Removal state, the REM_STAT bit is set and **p_enum_l** is asserted to indicate that a removal request is being made. Since the card is not yet ready for removal, the 21555 drives **l_stat** low in this state to force the LED off. When desired, the LED can be turned on by setting the LOO bit. Clearing the LOO bit causes the 21555 to drive **l_stat** low in this state.

After the software determines that the card is quiesced, or no longer performing or scheduling transactions, the host clears the REM_STAT bit. the 21555 deasserts **p_enum_l** and stops driving **l_stat** low. When the LOO bit is set, the 21555 continues to drive **l_stat** high. As long as the ejector handle stays open (**l_stat** sampled high), the LED will be on, indicating that it is OK to remove the card. The 21555 is now in the Removal OK state.

When **l_stat** is sampled low either upon clearing of REM_STAT, or anytime during the Removal OK state, this indicates that the ejector handle has been closed and the card is reseated. The 21555 enters the Signal Insertion state, setting the INS_STAT bit and asserting **p_enum_l**. Since no local state has been lost, serial preload and Primary Lockout Reset Value is not performed.

Clocking 7

The 21555 supports two clock inputs, **p_clk** and **s_clk**. The signal **p_clk** corresponds to the primary interface and **s_clk** corresponds to the secondary interface. Both clocks must adhere to the PCI Local Bus specification.

The 21555 may operate in either synchronous or asynchronous mode. The 21555 starts in asynchronous mode during reset, but can switch to synchronous mode after reset when **pr_ad[4]** is sampled low during reset.

In asynchronous mode, **p_clk** and **s_clk** can be asynchronous to each other. They can have any phase relationship and can differ in frequency.

When the 21555 operates in synchronous mode, **p_clk** and **s_clk** must operate at the same frequency and have a fixed phase relationship. Operation in synchronous mode saves at least one clock cycle of latency for transactions crossing the bridge. In this mode, the skew between **p_clk** and **s_clk** rising edges should be no less than 2 ns and no more than 13 ns (for 66 MHz). Therefore, the **s_clk** rising edges should never come before **p_clk** rising edges, and **s_clk** rising edges should not follow **p_clk** rising edges by more than 13 ns.

7.1 Primary and Secondary PCI Bus Clock Signals

Table 20 describes the primary and secondary PCI bus clock and 66 MHz enable signals.

Table 20. Primary and Secondary PCI Bus Clock Signals (Sheet 1 of 2)

Signal Name	I/O	Description
p_clk	I	Primary interface PCI CLK. This signal provides timing for all transactions on the primary PCI bus. All primary PCI inputs are sampled on the rising edge of **p_clk**, and all primary PCI outputs are driven from the rising edge of **p_clk**. The 21555 operates in a frequency range from 0 MHz to 66 MHz in synchronous mode. In asynchronous mode the 21555 supports a clocking ratio (defined p_clk : s_clk or s_clk : p_clk) of a maximum ratio 2.5 : 1 with the upper frequency limit for either clock input being 66MHz.
p_m66ena	I	Primary interface at 66 MHz. Signal **p_m66ena** asserted high indicates that the primary interface is operating at 66 MHz.

Table 20. Primary and Secondary PCI Bus Clock Signals (Sheet 2 of 2)

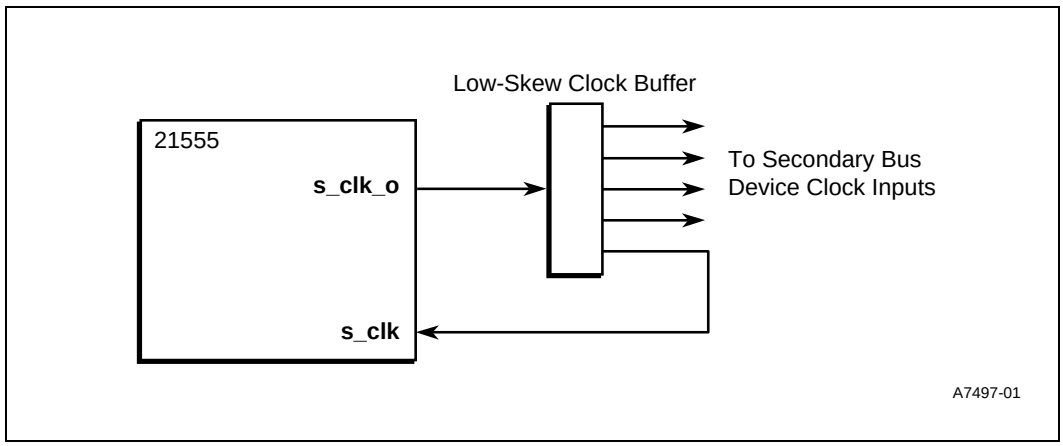
Signal Name	I/O	Description
s_clk	I	Secondary interface PCI CLK. This signal provides timing for all transactions on the secondary PCI bus. All secondary PCI inputs are sampled on the rising edge of **s_cl**k, and all secondary PCI outputs are driven from the rising edge of **s_cl**k. The 21555 operates in a frequency range from 0 MHz to 66 MHz in synchronous mode. In asynchronous mode the 21555 supports a clocking ratio (defined p_clk : s_clk or s_clk : p_clk) of a maximum ratio 2.5 : 1 with the upper frequency limit for either clock input being 66MHz
s_clk_o	O	Secondary interface PCI CLK output. Signal **s_clk_o** is a buffered version of **p_clk**. The 21555 divides **p_clk** by two to generate **s_clk_o** when **p_m66ena** is asserted high and **s_m66ena** is asserted low (the primary is operating at 66 MHz and the secondary is operating at 33 MHz). This signal is generated from the primary interface clock input, **p_clk**. This clock operates at the same frequency of **p_clk** and may be externally buffered to create secondary bus device clock signals. When buffered clocks are used, one of the clock outputs must be fed back to the secondary clock input, **s_clk**. This clock output can be disabled by writing the secondary clock disable bit in configuration space, or by pulling **pr_ad[5]** low during reset.
s_m66ena	I/OD	Secondary interface at 66 MHz. Signal **s_m66ena** asserted high indicates that the secondary interface is operating at 66 MHz. The 21555 pulls this signal down when the primary interface is operating at 33 MHz (**p_m66ena** low) and the secondary clock output **s_clk_o** is enabled.

7.2 21555 Secondary Clock Outputs

When the secondary clock is not supplied independently, the secondary clock output implemented on the 21555 can be used in either synchronous or asynchronous mode. The 21555 secondary clock output, **s_clk_o**, may be buffered externally for use with secondary bus devices and the 21555 secondary interface clock input, as shown in Figure 13. When **s_clk_o** is used for secondary bus devices, one of the externally buffered clock outputs must be used for the 21555 secondary clock input, **s_clk**. This clock output is a buffered version of **p_clk** and therefore has the same clock frequency as **p_clk**. An exception is when the primary bus is operating at 66 MHz and the secondary bus operates at 33 MHz, then the 21555 divides **s_clk_o** by 2 to generate a 33 Mhz clock (See Section 7.3).

Signal **s_clk_o** is disabled and driven low when the 21555 samples **pr_ad[5]** low during reset. Signal **s_clk_o** may also be disabled by setting the **s_clk_o** Disable bit in the Chip Control 0 configuration register.

Figure 13. Synchronous Secondary Clock Generation



7.3 66 MHz Support

The 21555 supports 66 MHz operation. It has two pins, **p_m66ena** and **s_m66ena**, that indicate whether the primary and secondary bus are operating at 66 MHz, respectively. Signal **p_m66ena** is an input-only pin.

- When sampled high, the primary bus is assumed to be operating at 66 MHz.
- When sampled low, the primary bus must be operating at or below 33 MHz. Signal **s_m66ena** is an input/ open-drain pin.
- When sampled high, the secondary bus is assumed to be operating at 66 MHz.
- When sampled low, the secondary bus must be operating at or below 33 MHz.

The 21555 pulls **s_m66ena** low when the primary bus is operating at 33 MHz (**p_m66ena** low) and **s_clk_o** is enabled. When **s_clk_o** is enabled, it is assumed that the 21555 is controlling the clocking of the secondary bus and since **s_clk_o** is a buffered version of **p_clk**, it must operate at
33 MHz.

When **p_m66ena** is sampled high, **s_m66ena** is sampled low, and **s_clk_o** is enabled, the 21555 divides **s_clk_o** by 2 to generate a 33Mhz clock.

The 21555 can handle any combination of clock frequencies between primary and secondary buses with the maximum clock ratio between primary and secondary buses being 2.5:1 (for example 25 MHz on one bus and 66 MHz on the other), and a maximum frequency of 66 MHz.

Parallel ROM Interface 8

This chapter presents the theory of operation information about the 21555 Parallel ROM (PROM) interface. See Chapter 16 for specific information about the PROM registers.

The 21555 supports the attachment of a standard PROM or EPROM with the addition of a small amount of external logic. Flash ROMs compatible with Intel's 28F00x can be used with this interface. The 21555 supports a PCI expansion ROM BAR on its primary interface with ROM sizes of 4KB to 16MB. Using these features the 21555 can provide the PCI expansion ROM interface for the subsystem. When the local subsystem does not require a PCI expansion ROM, the expansion ROM BAR can be disabled.

When the host completes the configuration of the primary PCI interface, the PROM may be accessed by the host in the memory address range assigned by the PCI expansion ROM BAR. The PROM is not directly accessible in the memory address space of the secondary PCI interface. However, the PROM can be indirectly accessed from both the primary and secondary PCI interfaces through the 21555 CSRs.

Note: The PROM cannot support simultaneous access using the Table 107, "Primary Expansion ROM BAR" on page 175 and the Table 112, "ROM Control Register" on page 178 at the same time. The results are unpredictable. Additionally, the ROM should not be accessed simultaneously from the primary and secondary interface using the Table 112, "ROM Control Register" on page 178, otherwise the results are unpredictable.

8.1 Interface Signals

The 21555 expansion ROM interface signals are listed in Table 21. The ROM address is driven out on the 8-bit data bus in three consecutive cycles. External octal D registers with active low enables are required to capture the ROM address.

The serial ROM data and clock signals are multiplexed with the ROM signals. A description of the serial ROM interface is given in Chapter 9. The PROM can also be used to interface other slave-only devices to the ROM address and data bus. This configuration is described in Section 8.7.

Table 21. PROM Interface Signals (Sheet 1 of 2)

Signal Name	Type	Description
pr_ad[7:0]	TS	These signals interface to both the serial and parallel external ROM circuitry and have multiple functions. The signals **pr_ad[7:0]** serve as multiplexed address/data for the PROM and are latched externally in the following sequence: • Address [23:16] during the first address cycle. • Address [15:8] during the second address cycle. • Address [7:0] during the third address cycle. • Data [7:0] during the data cycle. The signals **pr_ad[2:0]** also serve as serial ROM signals, with no external logic required: • **pr_ad[2]**: sr_do, the serial ROM data output. • **pr_ad[1]**: sr_di, the serial ROM data input. • **pr_ad[0]**: sr_ck, the serial ROM clock output. The value of **pr_ad[7:1]** signals during chip reset specifies the configuration options in the bit descriptions that follow. The values of these configuration options may be read from the Table 113, “Mode Setting Configuration Register” on page 179. **pr_ad[7]** Arbiter enable (active high). When low, the secondary bus arbiter is disabled, **s_gnt_l[0]** is used for 21555 secondary bus request, and **s_req_l[0]** is used for 21555 secondary bus grant. When high, the internal arbiter is enabled for use. **pr_ad[6]** Central function enable (active low). When low, the 21555 drives **s_a**d, **s_cbe**_l, and **s_par** low during secondary reset. When the secondary PCI interface is 64 bits, the 21555 also drives **s_req64_l** low. When high, the 21555 tristates s_req64_l, s_ad, s_cbe_l, and s_par during secondary reset. **pr_ad[5]** Signal **s_clk_o** enable (active high). When low, **s_clk_o** is turned off and driven low. When high, **s_clk_o** is turned on and is a buffered version of **p_clk**. **pr_ad[4]** Synchronous enable (active low). When high, the 21555 assumes asynchronous primary and secondary interfaces. When low, the 21555 assumes synchronous primary and secondary interfaces. **pr_ad[3]** Primary lockout bit reset value. When high, the primary lockout bit is set high upon completion of chip reset, causing the 2155X to return target retry to primary bus transactions until the bit is cleared. When low, the primary lockout bit is cleared low upon completion of reset, allowing immediate access to configuration registers. **pr_ad[2]** When the serial ROM is not connected, this pin should be pulled either high or low to disable the register preload. When the preload sequence 10b is not detected during the first read, the serial ROM preload is terminated after the first two bits are read and the 21555 registers remain at their reset values. This is not actually sampled at reset, but during the first serial ROM read. **pr_ad[1]** When the **s_rst_in_l** signal is used to reset the chip, sampling this signal low upon deassertion of **s_rst_in_l** enables the primary bus 64-bit extension. Sampling this signal high upon deassertion of **s_rst_in_l** disables the primary bus 64-bit extension, and those signals are then driven to valid logic values.

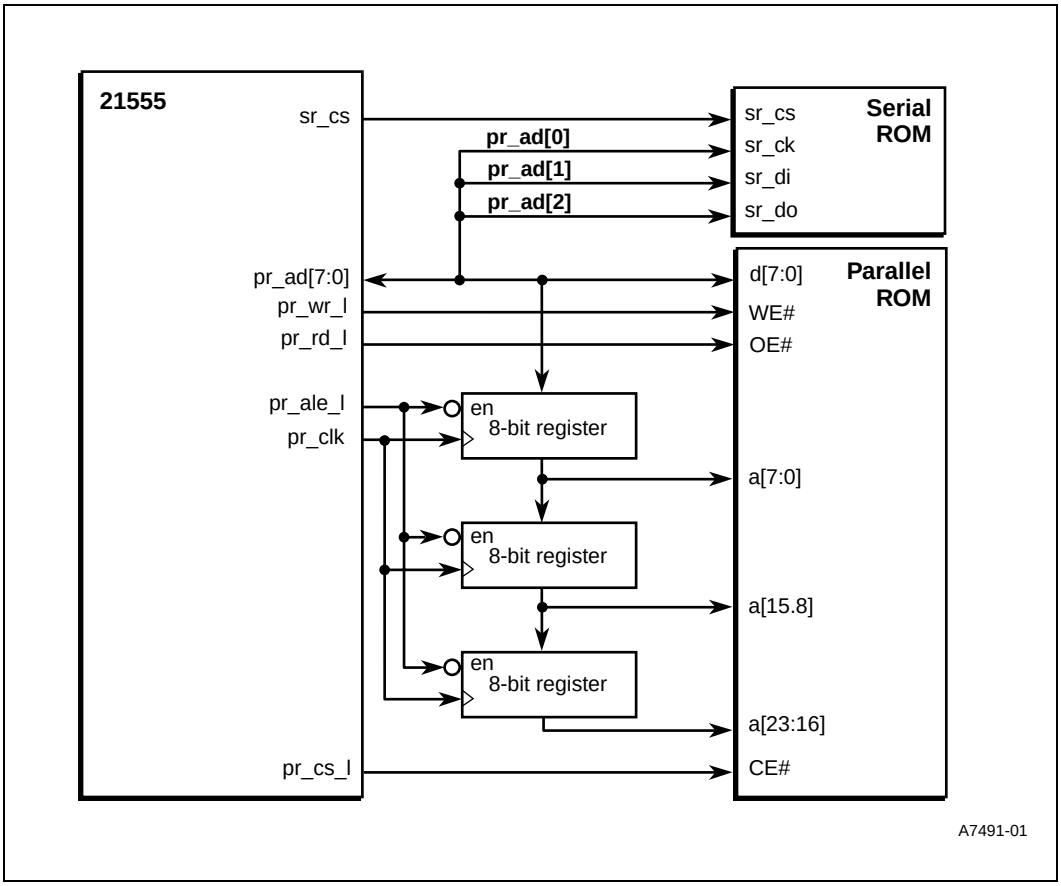
Table 21. PROM Interface Signals (Sheet 2 of 2)

Signal Name	Type	Description
pr_ale_l	O	PROM address latch enable/chip select decoder enable. The signal **pr_ale_l** is used to enable the PROM address latches. The 21555 asserts **pr_ale_l** low when it drives the first eight bits of the 24-bit address on **pr_ad[7:0]**, and keeps it asserted until the last eight bits of the address are driven. The address is shifted through three octal D-registers while **pr_ale_l** is low. When in multiple device mode, **pr_ale_l** is also used for a chip select enable. When **pr_ale_l** is high, the upper latched address lines are decoded with external circuitry to assert device chip enables.
pr_clk	O	PROM address latch clock output. The signal **pr_clk** is used to clock the three address registers needed to demultiplex the address. Signal **pr_clk** is divided by two when 33 MHz or **pr_clk** is divided by four when 66 MHz.
pr_cs_l/ pr_rdy	O/I	PROM chip select or device ready. For a single device attachment, **pr_cs_l** is used for the PROM chip select. The 21555 asserts **pr_cs_l** low after the address is shifted out and demultiplexer through the three external octal registers. The 21555 deasserts **pr_cs_l** according to the access time specified in the Table 112, “ROM Control Register” on page 178. When in multiple device mode, **pr_cs_l** is reconfigured as a device ready (pr_rdy) input. When **pr_cs_l** is driven low while the read or write strobe is asserted, the assertion time of the read or write strobe is extended by the amount of time the device ready signal is held low.
pr_rd_l	O	PROM read strobe. This signal controls the output enable signal of the PROM. The 21555 asserts **pr_rd_l** to enable the ROM to drive read data on **pr_ad[7:0]**. The 21555 samples this read data on the deasserting (rising) edge of **pr_rd_l**. The timing of **pr_rd_l** with respect to the chip select is dictated by the read strobe mask.
pr_wr_l	O	PROM write strobe. This signal controls the write enable signal of the PROM. The 21555 asserts **pr_wr_l** when it drives write data to the ROM on **pr_ad[7:0]**. Write data is held stable until the deasserting (rising) edge of **pr_wr_l**. The timing of **pr_wr_l** with respect to the chip select is dictated by the write strobe mask.
sr_cs	O	Serial ROM chip select. The 21555 drives this signal high to enable the serial ROM for a read or write. The serial ROM operation uses pins **pr_ad[2:0]** for data in, data out, and clock.

8.2 Parallel and Serial ROM Connection

Figure 14 shows how a parallel and serial ROM can be connected to the 21555. This figure illustrates the connection of a 16MB ROM. When a smaller ROM is used, the address registers corresponding to the upper address bits can be eliminated, as those upper address bits are ignored.

Figure 14. Parallel and Serial ROM Connections



8.3 PROM Read by CSR Access

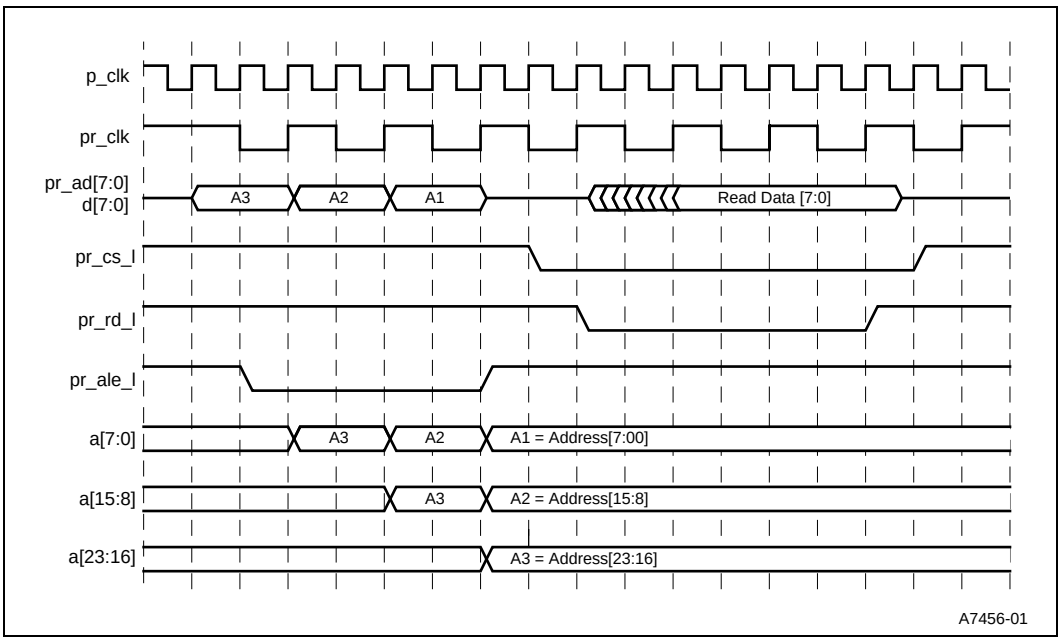
Byte reads of the PROM may be performed by CSR access of the ROM Control, Address, and Data registers. A byte read is performed as follows:

1. The initiator writes the byte address offset to the ROM Address register.
2. The initiator writes the PROM Start bit to a 1, Serial ROM Start bit to a 0, and the ROM Read/Write Control bit to a 0 in the Table 112, “ROM Control Register” on page 178.
3. When the initiator reads the PROM Start bit in the Table 112, “ROM Control Register” on page 178 as a 0, the ROM operation is complete and the initiator can obtain the read data from the ROM Data register.

When a byte read of the PROM is performed, the 21555 follows this sequence on the ROM interface, also shown in Figure 15.

1. The 21555 drives address bits [23:16] on the **pr_ad[7:0]** pins and asserts **pr_ale_l** to enable the address registers.
2. The 21555 drives **pr_clk** high, latching address bits [23:16] into the first external register.
3. The 21555 drives **pr_clk** low.
4. The 21555 drives address bits [15:8] on the **pr_ad[7:0]** pins.
5. The 21555 drives **pr_clk** high, latching address bits [15:8] into the first external register, and address bits [23:16] into the second external register.
6. The 21555 drives **pr_clk** low.
7. The 21555 drives address bits [7:0] on the **pr_ad[7:0]** pins.
8. The 21555 drives **pr_clk** high, latching address bits [7:0] into the first external register, address bits [15:8] into the second external register, and address bits [23:16] into the third external register. All the ROM address bits are now driven to the appropriate ROM pins.
9. The 21555 deasserts the address register enable, **pr_ale_l**.
10. The 21555 asserts the **pr_cs_l** and **pr_rd_l** pins according to the strobe setup timing specified by the Strobe Mask in the ROM Setup register.
11. The PROM drives read data onto the **pr_ad[7:0]** pins.
12. The 21555 samples the read data and deasserts **pr_rd_l** as specified by the strobe mask. The 21555 also deasserts **pr_cs_l** according to the access time specified in the ROM Setup register.
13. The 21555 clears the PROM Start bit in the Table 112, “ROM Control Register” on page 178.
14. Valid data can now be read from the ROM Data register.

Figure 15. PROM Read Timing



8.4 PROM Write by CSR Access

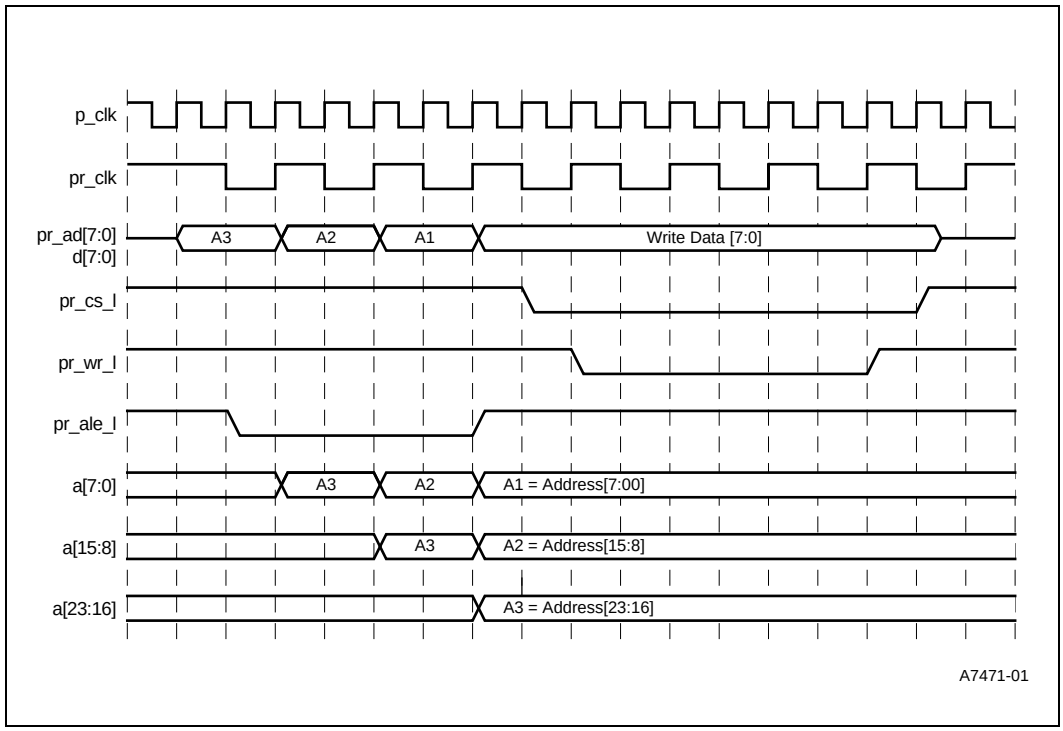
Byte writes of the PROM can be performed by CSR access of the Table 112, "ROM Control Register" on page 178, Table 111, "ROM Address Register" on page 178, and Table 110, "ROM Data Register" on page 177. A byte write is performed as follows:

1. The initiator writes the byte address offset to the ROM Address register.
2. The initiator writes one byte of write data into the ROM Data register.
3. The initiator writes the PROM Start bit to a 1, Serial ROM Start bit to a 0, and the ROM Read/Write Control bit to a 1 in the Table 112, "ROM Control Register" on page 178 This can be done with the same CSR access.
4. When the initiator reads the PROM Start bit in the Table 112, "ROM Control Register" on page 178 as a 0, the access is complete.

When a byte write to the PROM is performed, the 21555 follows this sequence on the ROM interface, also shown in Figure 16:

1. The 21555 drives address bits [23:16] on the **pr_ad[7:0]** pins and asserts the address register enable, **pr_ale_l**.
2. The 21555 drives **pr_clk** high, latching address bits [23:16] into the first external register.
3. The 21555 drives **pr_clk** low.
4. The 21555 drives address bits [15:8] on the **pr_ad[7:0]** pins.
5. The 21555 drives **pr_clk** low, latching address bits [15:8] into the first external register, and address bits [23:16] into the second external register.
6. The 21555 drives **pr_clk** low.
7. The 21555 drives address bits [7:0] on the **pr_ad[7:0]** pins.
8. The 21555 drives **pr_clk** high, latching address bits [7:0] into the first external register, address bits [15:8] into the second external register, and address bits [23:16] into the third external register. All the ROM address and control bits are now driven to the appropriate ROM pins.
9. The 21555 deasserts the address register enable, **pr_ale_l**.
10. The 21555 drives the write data on **pr_ad[7:0]**.
11. The 21555 asserts the **pr_cs_l** and **pr_wr_l** pins according to the strobe setup timing specified by the Strobe Mask in the ROM Setup register.
12. The 21555 deasserts **pr_wr_l** according to the strobe timing in the ROM Setup register, and deasserts the **pr_cs_l** according to the access time in the ROM Setup register.
13. The 21555 clears the PROM Start bit in the Table 112, "ROM Control Register" on page 178.

Figure 16. PROM Write Timing



8.5 PROM Dword Read

A Dword read is performed on the PROM interface when a read is initiated on the primary bus whose address falls into the address range defined by the Table 107, “Primary Expansion ROM BAR” on page 175. The 21555 treats a memory read through the Table 107, “Primary Expansion ROM BAR” on page 175 as a delayed read and returns a target retry to the initiator. The 21555 performs four consecutive byte reads of the ROM. When the four byte reads are complete, the 21555 returns the read data to the initiator on the next read attempt to that address to complete the delayed transaction. The 21555 automatically sets the PROM Start/Busy bit upon initiation of the read and clears the bit when the ROM read is complete.

Note: The 21555 uses the ROM Address CSR to hold for comparison the address decoded in the Primary Expansion ROM address space. The CSR access method should not be used for the PROM when reads to the PROM through the Table 107, “Primary Expansion ROM BAR” on page 175 are taking place, because the address for the CSR access can become corrupted by a Expansion ROM BAR read.

The delayed transaction mechanism for the expansion ROM BAR does not support multiple masters. Assume the 21555 completes a ROM Dword read on the ROM interface, but before the data is returned to the master a transaction using a different offset in the expansion ROM BAR range is initiated. The 21555 discards the current completed transaction and queues the second transaction. When the first transaction had not been completed on the ROM interface, a target retry would have been returned.

8.6 Access Time and Strobe Control

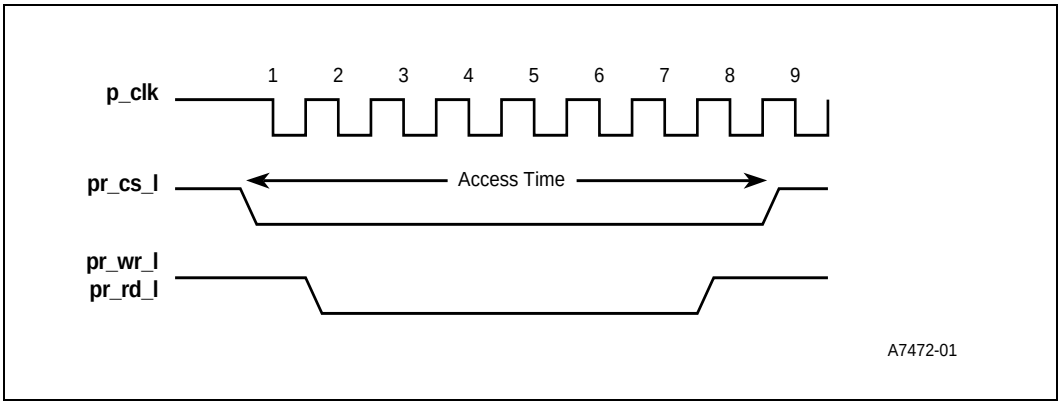
The 21555 controls both the access time and the read and write strobe timing through the ROM Setup CSR.

The access time is specified as a multiple of the **p_clk** signal and must be set to 8, 16, 64, or 256 times the length of a **p_clk** cycle when **p_clk** is operating at 33 MHz or below, and 16, 32,128, or 512 times the length of a **p_clk** cycle when **p_clk** is operating above 33 MHz. This specifies the number of **p_clk** cycles that the 21555 asserts **pr_cs_l**. The reset value is 8 times the 33 MHz **p_clk** cycle time or 16 times the 66 MHz **p_clk** cycle time.

The read and write strobe timing of **pr_rd_l** and **pr_wr_l** is given as an 8-bit mask. Each bit corresponds to one eighth of the access time, which can be 1, 2, 8, or 32 **p_clk** cycles when **p_clk** is operating at 33 MHz or below and 2, 4, 16, or 64 **p_clk** cycles when **p_clk** is operating above 33MHz. Bit 0 corresponds to the first cycle. When a bit is a 0 (zero), the read or write strobe is deasserted. When the bit is a 1, the read or write strobe is asserted. Signal **pr_cs_l** is asserted at the beginning of the first cycle and deasserted at the end of the last cycle. The reset value for the strobe mask is 01111110b. Since the reset value of the access time is either 8 each 33 MHz **p_clk** cycles or 16 each 66 MHz **p_clk** cycles, this means a read or write access has the following timing (at 33 MHz), also shown in Figure 17:

- Cycle 1: assert **pr_cs_l.**
- Cycle 2 through 7: both **pr_cs_l** and **pr_rd_l** or **pr_wr_l** asserted.
- Cycle 8: deassert **pr_rd_l** or **pr_wr_l.**
- Cycle 9: deassert **pr_cs_l**.

Figure 17. Read and Write Strobe Timing



8.7 Attaching Additional Devices to the ROM Interface

The 21555 allows additional devices to be attached to the ROM interface. Two ROM interface signals are slightly redefined to support multiple devices by setting the Multiple Device Enable bit in the Chip Control 0 configuration register. In this mode, the maximum ROM size is reduced because the upper address lines are used to decode device select lines.

For this mode of operation, an external decoder is needed to decode the upper address bits into device select lines. The **pr_ale_l** signal is used to enable the device select decoder, in addition to enabling the clocking of the address registers. When **pr_ale_l** is low, the address registers are enabled, and when **pr_ale_l** is high, the device select decoder is enabled.

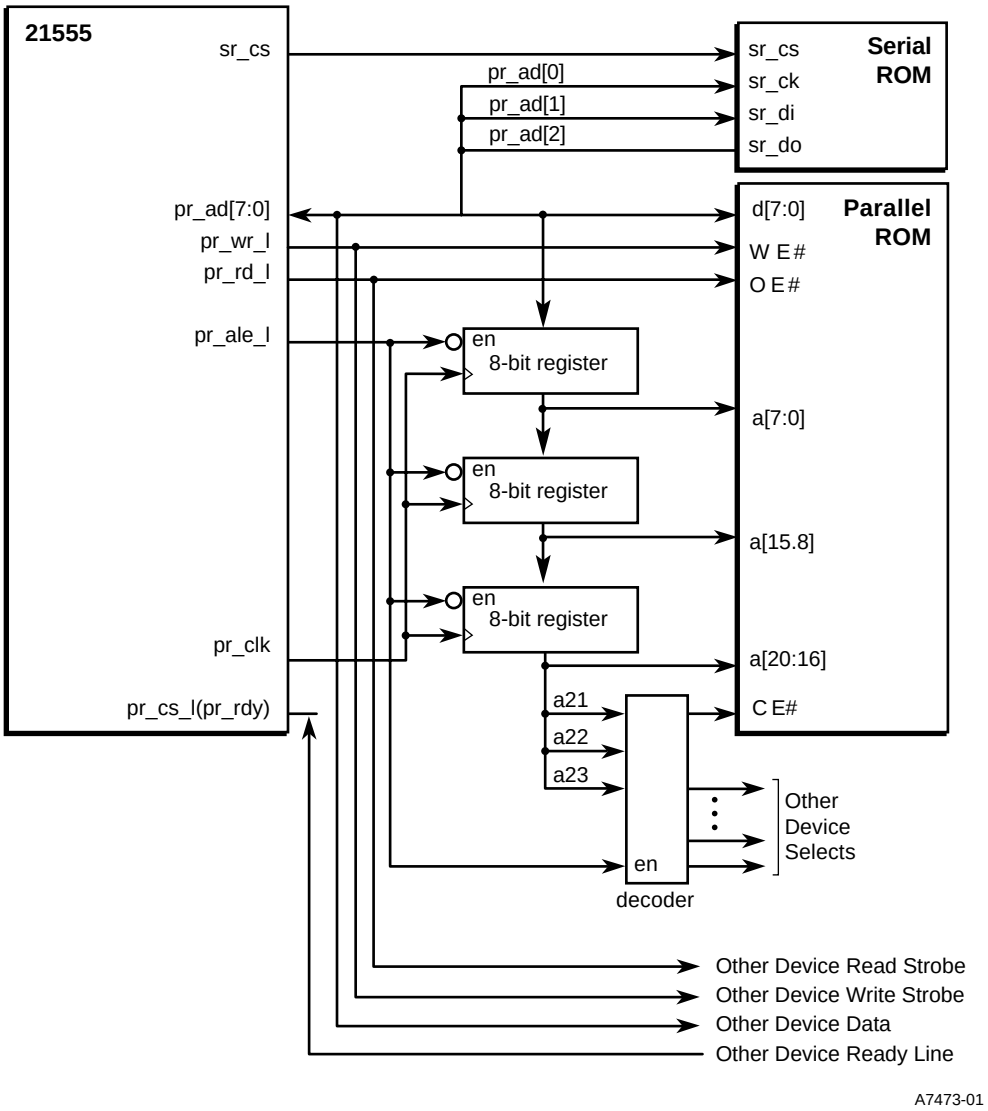
> ***Note:*** Signal **pr_ale_l** must be driven low when the serial ROM is used to disable all other device select lines when an external decoder is used.

In addition, in this mode the **pr_cs_l** output pin is redefined to be a device ready input (**pr_rdy**). When **pr_rdy** is deasserted, the device select signal (controlled by **pr_ale_l**) and read or write strobe assertions are extended until pr_rdy is asserted again. The read and write strobes are deasserted upon detection of chip select deassertion (fall time for **pr_ale_l**) with the hold time specified by the strobe mask.

> ***Note:*** When multiple devices are attached and multiple device mode is used, signal **pr_cs_l** should be pulled up through an external resistor.

Figure 18 illustrates how the interface is connected for use with multiple devices.

Figure 18. Attaching Multiple Devices on the ROM Interface



Serial ROM Interface 9

This chapter presents the theory of operation information about the 21555 Serial ROM (SROM) interface. See Chapter 16 for specific information about the SROM registers.

The serial ROM interface is used to preload data into the 21555 configuration registers with vendor-specific values. The format for the serial ROM data is given in Section 9.3. The SROM can be support the Vital Product Data (VPD) interface as described in Chapter 15.

The SROM interface works with the Microchip 93LC66A* or compatible SROM, which is a byte-organized Microwire SROM with 4 Kb (512 bytes) of storage. The clock input to the SROM is the primary clock input, **p_clk**, operating at a maximum clock frequency of 33 MHz divided by 34. When **p_clk** is operating above 33 MHz, it is divided by 68 to generate the SROM clock input. The duty cycle is approximately 50%.

9.1 SROM Interface Signals

The SROM interface consists of four signals, as shown in Table 22. The chip select, **sr_cs**, has a dedicated pin. The other signals are multiplexed with the PROM (PROM) interface signals (refer to Table 21). The SROM may be attached directly to the SROM pins without additional external logic.

Table 22. SROM Interface Signals

Name	Type	Description	21555 Pin
sr_cs	O	Serial ROM Chip Select	sr_cs
sr_ck	O	Serial ROM Clock	pr_ad[0]
sr_di	O	Serial ROM Data In	pr_ad[1]
sr_do	I	Serial ROM Data Out	pr_ad[2]

9.2 SROMSROM Preload Operation

The SROM interface is used to preload the 21555 configuration registers whenever the 21555 configuration registers are reset, either through assertion of **p_rst_l** or **s_rst_l**, by setting the Chip Reset bit in the Chip Control Register or after a power management transition from $D3_{hot}$ to D0.

Once reset is complete, the 21555 automatically starts a serial read from the ROM by detecting that **p_rst_l** and **s_rst_l** are deasserted and the Chip Reset bit reset to 0 (zero). All of the 21555 initialization data is loaded with a single read operation by keeping the chip select asserted and toggling the clock. The 21555 returns a target retry to all configuration accesses until the preload operation is complete. The preload operation takes approximately 550 SROM clock cycles.

When the SROM is not present, the **sr_do** (pin pr_ad[2]) should be pulled up through an external resistor. When the SROM is present but register preload is not desired, bits [7:6] of the first byte (the first two bits read) can be any value except the preload enable sequence 10b. When the 21555 does not detect the preload enable sequence when reading the first byte, it stops the preload operation. In this case, all configuration registers preloaded with the SROM remain at their reset value and should be initialized by the local processor before host access.

9.3 SROM Configuration Data Preload Format

Some fields of the 21555 configuration registers may be preloaded using the SROM interface. The first two bits read from the SROM after the completion of chip reset indicate whether a register preload should be performed. When the first two bits read as 10b, an auto-load sequence is initiated. The SROM is sequentially read and the data is shifted along a scan register chain to load the registers listed in Table 114, “Serial Preload Sequence” on page 180.

9.4 SROM Operation by CSR Access

The 21555 allows SROM access through CSR control. A SROM operation consists of the following three phases:

1. Command phase of 3 bits
2. Address phase of 9 bits
3. Data phase of 8 or more bits
   - — Read operations may consist of any number of data bits.
   - — Write operations always consist of 8 data bits.

For a read type operation, the data is driven from the SROM to the 21555 on signal **sr_do**. For a write operation, the data is driven from the 21555 to the SROM on signal **sr_di**.

To perform a SROM access, the initiator should make sure that both the parallel and SROM Start/Busy bits are clear in the ROM Control CSR. The SROM byte address and the SROM opcode are then written to the ROM Address CSR. Defined opcodes are:

00	Write enable, write disable, write all, erase all
01	Write
10	Read
11	Erase

For opcode 00, a byte address is not used and the two most significant address bits [8:7] distinguish between the four commands:

00	Write disable
01	Write all
10	Erase all
11	Write enable

Prior to a SROM write or write all transaction, the 8-bit write data must be written in the ROM Data CSR.

To initiate the SROM access, the SROM Start bit in the ROM Control CSR is written with a 1 (the PROM Start bit must be written to a 0 with this access). The 21555 then initiates the SROM access. When the SROM access is complete, the 21555 automatically clears the SROM Start bit. When the operation is a read, the data then can be read from the ROM Data register.

The write, write all, erase, and erase all commands may take 10 ms or more to complete, internal to the ROM. A poll of the SROM must be performed to discover whether these operations are complete. For these commands, when the SROM access is initiated, the 21555 also sets the SROM_POLL bit in the Table 112, “ROM Control Register” on page 178. This bit remains asserted after the 21555’s access to the SROM completes. The SROM must be polled by CSR access and return a ready indication to clear the SROM_POLL bit.

The SROM is polled by the 21555 when the SROM Start bit is written with a 1 when the SROM_POLL bit is set. The 21555 asserts **sr_cs** and drives **sr_di** (pin pr_ad[1]) low. When the SROM drives **sr_do** (pin pr_ad[2]) high in response, it has completed the operation internally and the 21555 clears the SROM_POLL bit. The SROM is now ready for another access.

> ***Note:*** The SROM_POLL bit must be set for the 21555 to poll the SROM, otherwise the 21555 initiates another SROM access if the SROM Start bit is written.

A summary of the actions needed for a SROM read access follows:

1. The initiator writes the byte address and the opcode in the ROM Address CSR.
2. The initiator writes the SROM Start bit to a 1 and the PROM Start bit to a 0 in the ROM Control CSR.
3. When the SROM Start bit in the ROM Control CSR is read as a zero, the initiator may read the 8-bit data from the ROM Data register.

A summary of the actions needed for a write operation follows:

1. The initiator writes the byte address and the opcode in the ROM Address CSR.
2. The initiator writes the 8-bit data in the ROM Data CSR.
3. The initiator writes the SROM Start bit to a 1 and the PROM Start bit to a 0 in the ROM Control CSR in the same CSR access.
4. When the SROM Start bit in the ROM Control CSR is read as a zero, the initiator polls the SROM to test for write completion by writing the SROM Start bit to a 1.
5. When the SROM Start bit in the ROM Control CSR is read as a zero, the SROM_POLL bit indicates the status of the polling operation. When SROM_POLL is read as a one, the SROM should be polled again. When SROM_POLL is read as a 0, the operation is complete.

The erase, erase all, write enable, and write disable all use write protocol. For all of these operations, however, the ROM Data register does not need to be written. In addition, the write enable and write disable operations do not require polling for completion. Figure 19 through Figure 24 show the timing diagrams for SROM read, write, write all, write enable, write disable, erase, erase all, and check status (polling) operations.

Note: When a SROM access using the CSR mechanism is attempted when the SROM is not implemented, the ROM interface may hang. This prevents access to any PROMs that may be present. A chip reset may be needed to put the ROM interface in an operational state

Figure 19. SROM Write All Timing Diagram

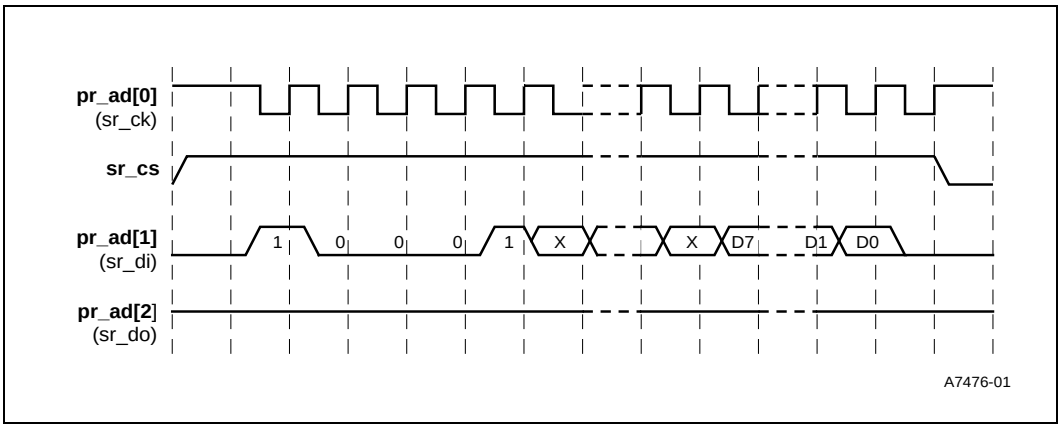


Figure 20. SROM Write Enable Timing Diagram

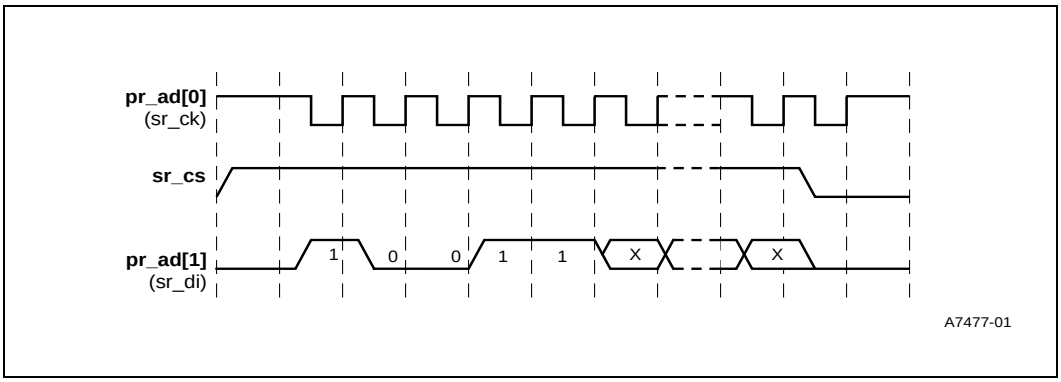


Figure 21. SROM Write Disable Timing Diagram

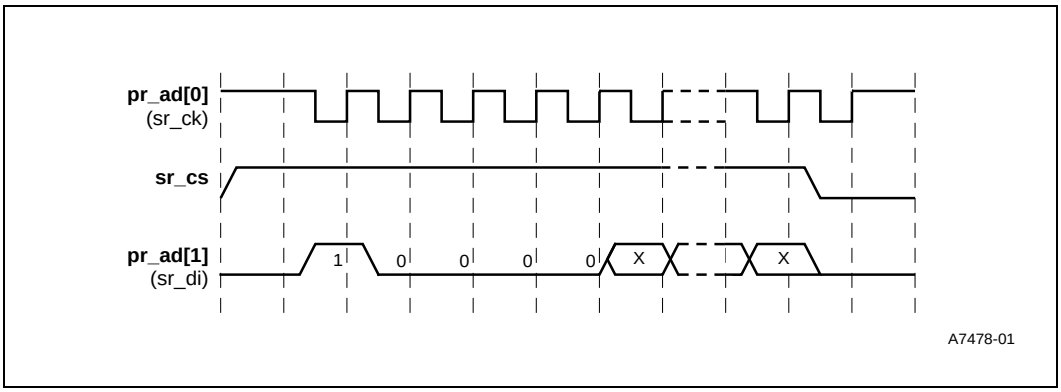


Figure 22. SROM Erase Timing Diagram

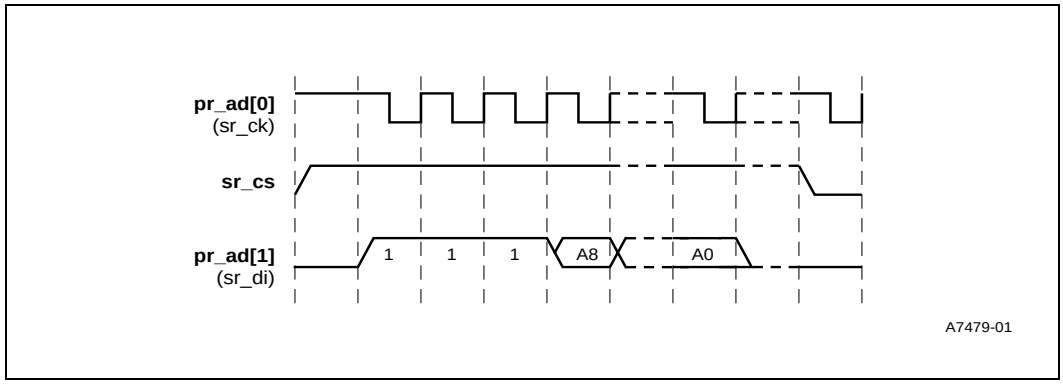


Figure 23. SROM Erase All Operation

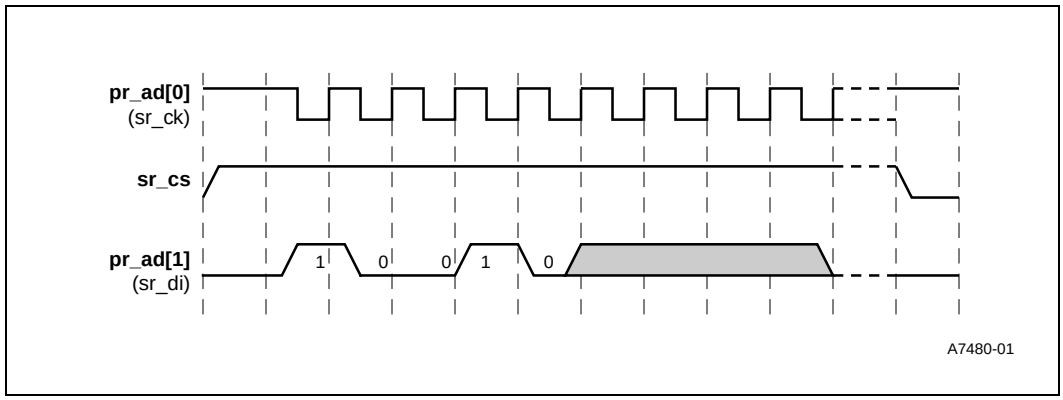
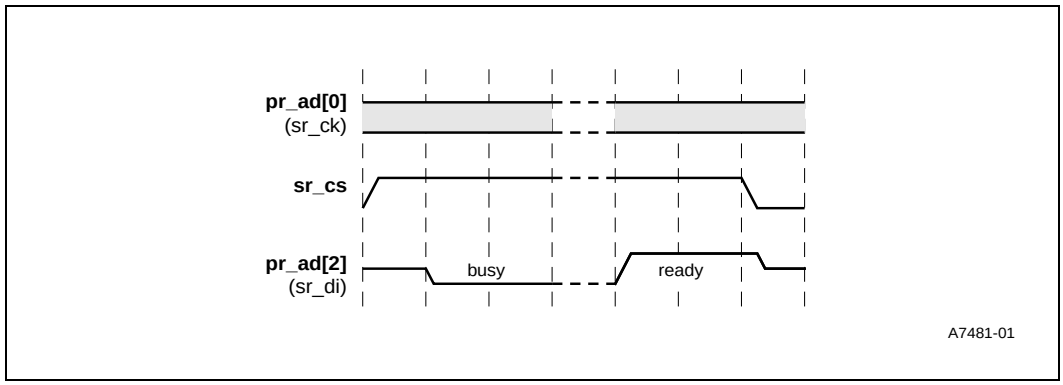


Figure 24. SROM Check Status Timing Diagram



Arbitration 10

This chapter describes the arbitration signals. It also describes how the 21555 implements primary and secondary PCI bus arbitration. See Chapter 16 for specific information about the Arbiter registers.

10.1 Primary PCI Bus Arbitration Signals

Table 23 describes the primary PCI bus arbitration signals.

Table 23. Primary PCI Bus Arbitration Signals

Signal Name	Type	Description
p_gnt_l	I	Primary PCI bus GNT#. When asserted, **p_gnt_l** indicates to the 21555 that access to the primary bus is granted. The 21555 can start a transaction on the primary bus when the bus is idle and **p_gnt_l** is asserted. When the 21555 has not requested use of the bus and **p_gnt_l** is asserted, the 21555 drives **p_ad**, **p_cbe_l**, and **p_par** to valid logic levels.
p_req_l	TS	Primary PCI bus REQ#. Signal **p_req_l** is asserted by the 21555 to indicate to the primary bus arbiter that it wants to start a transaction on the primary bus.

10.2 Secondary PCI Bus Arbitration Signals

Table 24 describes the secondary PCI bus arbitration signals.

Table 24. Secondary PCI Bus Arbitration Signals

Signal Name	Type	Description
s_gnt_l[8:0]	TS	Secondary PCI interface GNT#s. The 21555 secondary bus arbiter can assert one of nine secondary bus grant outputs, **s_gnt_l[8:0]**, to indicate that an initiator can start a transaction on the secondary bus if the bus is idle. The 21555's secondary bus grant is an internal signal. A programmable two-level rotating priority algorithm is used. When the internal arbiter is disabled, **s_gnt_l[0]** is reconfigured to be an external secondary bus request output for the 21555. The 21555 asserts this signal whenever it wants to start a transaction on the secondary bus.
s_req_l[8:0]	I	Secondary PCI interface REQ#s. The 21555 accepts nine request inputs, **s_req_l[8:0]**, into its secondary bus arbiter. The 21555's request input to the arbiter is an internal signal. Each request input can be programmed to be in either a high-priority rotating group or a low-priority rotating group. An asserted level on an **s_req_l** pin indicates that the corresponding master wants to initiate a transaction on the secondary PCI bus. When the internal arbiter is disabled, **s_req_l[0]** is reconfigured to be an external secondary grant input for the 21555. In this case, an asserted level on **s_req_l[0]** indicates that the 21555 can start a transaction on the secondary PCI bus when the bus is idle.

10.3 Primary PCI Bus Arbitration

The 21555 implements primary PCI bus request and grant pins, **p_req_l** and **p_gnt_l**, that interface to an external primary bus arbiter. These pins are used when the 21555 wants to initiate a transaction on the primary PCI bus.

The 21555 asserts **p_req_l** when a posted write or delayed transaction is queued in upstream buffers. Signal **p_req_l** remains asserted as long as the posted write and delayed transaction queues contain pending transactions; otherwise, **p_req_l** is deasserted two cycles after the address phase. However, when the 21555 keeps **p_req_l** asserted and the 21555 detects a target retry or target disconnect in response to an ongoing transaction, it deasserts **p_req_l** one cycle after detecting that **p_stop_l** is asserted before reattempting arbitration for that transaction. The signal **p_req_l** is deasserted for two clock cycles.

When a prefetchable read is ongoing on the primary bus when another delayed read is queued behind it, the 21555 delays the assertion of **p_req_l**. The assertion of **p_req_1** is delayed until the 21555 is ensured that there is room in the read data queue for the second delayed read transaction.

When **p_gnt_l** is asserted when **p_req_l** is not asserted, the 21555 parks **p_ad**, **p_cbe_l**, and **p_par** by driving them to valid logic levels. The 64-bit extension signals are not parked. When the primary bus is parked at the 21555, and the 21555 has a transaction to initiate on the primary bus, it starts the transaction immediately as long as **p_gnt_l** was asserted during the previous clock cycle.

10.4 Secondary PCI Bus Arbitration

The 21555 implements an internal secondary PCI bus arbiter supporting nine secondary bus masters, plus the 21555. The internal arbiter may be disabled and an external arbiter used for secondary bus arbitration.

The behavior of the 21555 secondary request is identical to the behavior of the 21555's primary bus request.

10.4.1 Secondary Bus Arbitration Using the Internal Arbiter

The 21555 enables the secondary bus arbiter when it detects **pr_ad[7]** high during reset. The 21555 has nine secondary bus request input pins, **s_req_l[8:0]**, and nine secondary bus output grant pins, **s_gnt_l[8:0]**, to support external secondary bus masters. The 21555 secondary bus request and grant signals are connected internally to the arbiter and are not brought out to external pins when the arbiter is enabled. The minimum latency between secondary bus request assertion and secondary bus grant assertion is two clock cycles.

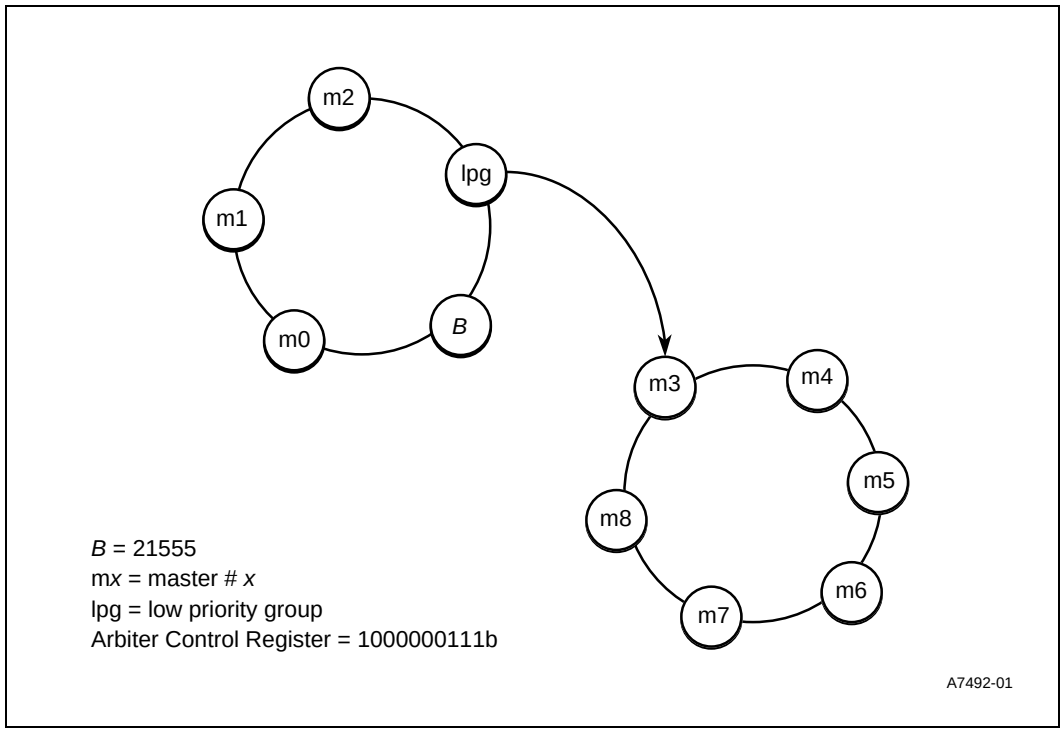
The secondary arbiter supports a programmable two level rotating priority algorithm. Two groups of masters are assigned, a high priority group and a low priority group. The low priority group as a whole represents one entry in the high priority group. That is, when the high priority group consists of N masters, then in at least every N+1 transactions the highest priority is assigned to the low priority group. Priority rotates evenly among the low priority group. Therefore, members of the high priority group can be serviced N transactions out of N+1, while one member of the low priority group is serviced once every N+1 transactions.

Figure 25 is an example where four masters, including the 21555, are in the high priority group and six masters are in the low priority group.

When all requests are asserted, the highest priority rotate among the masters in the following fashion (high priority members in italics, low priority members in bold):

B, m0, m1, m2, **m3,** *B, m0, m1, m2,* **m4,** *B, m0, m1, m2,* **m5,** *B, m0, m1, m2,* **m6,** *B, m0, m1,...* etc.

Figure 25. Secondary Arbiter Example



Each bus master, including the 21555, may be configured to be in either the low priority group or the high priority group by setting the corresponding priority bit in the Arbiter Control register in device-specific configuration space. When the bit is set to a one, the master is assigned to the high priority group. When the bit is set to a zero, the master is assigned to the low priority group. When all the masters are assigned to one group, the algorithm defaults to a straight rotating priority among all the masters. After reset, all external masters are assigned to the low priority group and the 21555 is assigned to the high priority group. The 21555 receives highest priority on the target bus every other transaction, and priority rotates evenly among the other masters.

Priorities are reevaluated every time **s_frame_l** is asserted, at the start of each new transaction on the secondary PCI bus. From this point until the time that the next transaction starts, the arbiter asserts the grant signal corresponding to the highest priority request that is asserted. When a grant for a particular request is asserted and a higher priority request subsequently asserts, the arbiter deasserts the asserted grant signal and asserts the grant corresponding to the new higher priority request on the next PCI clock cycle. The 21555 allocates a two-cycle minimum assertion time during bus idle once a grant is asserted to a bus master. When priorities are reevaluated, the highest priority is assigned to the next highest priority master relative to the master that initiated the previous transaction. The master that initiated the last transaction has the lowest priority in its group.

When the 21555 detects that a master has failed to assert **s_frame_l** after 16 cycles of both grant assertion and a secondary idle bus condition, the arbiter deasserts the grant. That master does not receive any more grants until it deasserts its request for at least one PCI clock cycle.

To prevent bus contention, when secondary **FRAME#** is deasserted, the arbiter does not assert one grant signal in the same PCI cycle as it deasserts another. It deasserts one grant, and then asserts the next grant no earlier than one PCI clock cycle later. When **s_frame_l** is asserted, the arbiter can deassert one grant and assert another grant during the same PCI clock cycle.

The 21555's internal arbiter may be programmed to park the secondary PCI bus either at the last master to use the bus, or always on the 21555. In the former case, an initiator's secondary bus grant remains asserted unless and until another initiator has asserted its secondary bus request. In the latter case, when no requests are asserted once a transaction has been initiated, the bus grant is withdrawn from the last master and is asserted internally to the 21555. After reset, the internal arbiter always parks the secondary bus at the 21555.

For secondary bus internal arbitration, 21555 internal arbitration signal pairs [5:8] are disabled for 66 MHz operation.

10.4.2 Secondary Bus Arbitration Using an External Arbiter

The internal arbiter is disabled when **pr_ad[7]** is detected low during reset. An external arbiter must then be used. When the internal arbiter is disabled, the 21555 redefines two pins to be external request and grant pins. The **s_gnt_l[0]** pin is redefined to be the 21555's external request pin, since it is an output. The **s_req_l[0]** pin is redefined to be the external grant pin, since it is an input. The unused secondary bus grant outputs, **s_gnt_l[8:1]**, are driven high. Unused secondary bus request inputs, **s_req_l[8:1]**, should be pulled high through external resistors.

When **s_req_l[0]** is asserted and the 21555 has not asserted **s_gnt_l[0]**, the 21555 parks the **s_ad**, **s_cbe_l**, and **s_par** pins by driving them to valid logic levels. The 64-bit extension signals on the 21555 are not bus parked.

Table 25. Arbiter Control Register

Bit	Name	R/W	Description
9:0	Arbiter Control	R/W	Each bit controls whether a secondary bus master is assigned to the high priority arbiter ring or the low priority arbiter ring. Bits [8:0] correspond to request inputs **s_req_l[8:0]**, respectively. Bit [9] corresponds to the internal 21555 secondary bus request. When 0, Indicates that the master belongs to the low priority group. *When 1:* Indicates that the master belongs to the high priority group. *Reset value:* 10 0000 0000b.
10	Bus Parking Control	R/W	Controls whether the 21555 parks on itself or on the last master to use the bus. When 0, During bus idle, the 21555 parks the bus on the last master to use the bus. *When 1:* During bus idle, the 21555 parks the bus on itself. The bus grant is removed from the last master and internally asserted to the 21555. *Reset value:* 0b.
15:11	Reserved	R	Reserved. Returns 0 when read.

Interrupt and Scratchpad Registers 11

This chapter presents the theory of operation information about the 21555 interrupt handling and about the 32-bit scratchpad registers. See Chapter 16 for specific information about these registers.

11.1 Primary and Secondary PCI Bus Interrupt Signals

Table 26 describes the primary and secondary PCI bus interrupt signals.

Table 26. Primary and Secondary PCI Bus Interrupt Signals

Signal Name	Type	Description
p_inta_l	OD	Primary PCI bus interrupt. Signal **p_inta_l** is asserted by the 21555 when: A primary doorbell register bit is set. The I20 outbound queue is not empty. The subsystem event bit is set. All of these conditions are individually maskable. When the corresponding event bit is cleared or the I20 outbound queue is emptied, **p_inta_l** is deasserted. Signal **p_inta_l** is pulled up through an external resistor.
s_inta_l	OD	Secondary PCI bus interrupt. Signal **s_inta_l** is asserted by the 21555 when: A secondary doorbell register bit is set. The I20 inbound queue is not empty. A page boundary is reached when performing lookup table address translation. The 21555 transitions from a D1 or D2 power state to a D0 power state. All of these conditions are individually maskable. Signal **s_inta_l** is deasserted when the corresponding event bit is cleared, or when the I20 inbound queue is empty. Signal **s_inta_l** is pulled up through an external resistor.

11.2 Interrupt Support

The 21555 supports hardware to facilitate software-generated interrupts, as well as interrupts initiated by the 21555 activity. The 21555 has interrupt request status and interrupt mask bits for the following conditions:

- I20 Inbound Post_List FIFO not empty.
  - — Cleared automatically when FIFO is empty.
  - — Asserts **s_inta_l** when the corresponding mask bit is zero.
- I20 Outbound Post_List FIFO not empty.
  - — Cleared automatically when FIFO is empty.
  - — Asserts **p_inta_l** when the corresponding mask bit is zero.
- Upstream memory read or write Dword transfer in Upstream Memory Range 2 addresses the last Dword in a page.
  - — One event and enable bit for each of the 64 pages in Upstream Memory Range 2.

  - — Cleared by writing a 1 to the corresponding status bit in the Upstream Page Boundary IRQ 0 or 1 registers.
  - — Asserts **s_inta_l** when the corresponding mask bit is zero.
- A subsystem event is indicated by a rising edge on **s_pme_l**.
  - — Cleared by writing a 1 to the corresponding status bit in the Chip Status CSR.
  - — Asserts **p_inta_l** when the corresponding mask bit is zero.
- A power management transition from state D1 or D2 to state D0 occurs.
  - — Cleared by writing a 1 to the corresponding status bit in the Chip Status CSR.
  - — Asserts **s_inta_l** when the corresponding mask bit is zero.

11.3 Doorbell Interrupts

A 16-bit software controlled interrupt request register and an associated 16-bit mask register is implemented for each interface (primary and secondary). Each register is byte addressable for use as two sets of 8-bit interrupt request and interrupt mask registers for each interface (four in all) if desired. These registers can be accessed from the primary or secondary interface of the 21555, in either memory space or I/O space.

The 21555 doorbell interrupt functionality consists of the following registers:

- Primary Interrupt Request 16-bit register.
- Secondary Interrupt Request 16-bit register.
- Primary Interrupt Request (IRQ) Mask 16-bit register.
- Secondary Interrupt Request (IRQ) Mask 16-bit register.

The primary interrupt pin, **p_inta_l**, is asserted low whenever one or more Primary Interrupt Request bits are set and their corresponding Primary IRQ Mask bits are 0. Signal **p_inta_l** remains asserted as long as this condition exists. Signal **p_inta_l** is deasserted when either the Primary Interrupt Request bit is cleared or the Primary IRQ Mask bit is set. The secondary interrupt pin, **s_inta_l**, is asserted low when one or more Secondary Interrupt Request bits are set and their corresponding Secondary IRQ Mask bits are 0 (zero), and remains asserted as long as this condition exists. The signal **s_inta_l** is deasserted when either the Secondary Interrupt Request bit is cleared or the Secondary IRQ Mask bit is set.

Each register can be accessed at two addresses. One location is used to set bits and the other location is used to clear them. To modify a request bit, a 1 is written to the bit in either the write-1-to-set interrupt or write-1-to-clear interrupt register address. Interrupt status can be read from either register.

11.4 Scratchpad Registers

Table 106, “Scratchpad 0 Through Scratchpad 7 Registers” on page 174 lists the bit descriptions of the scratchpad 0 through scratchpad 7 registers.

The eight 32-bit scratchpad registers can be accessed in either memory or I/O space from either the primary or secondary interface. They can pass control and status information between primary and secondary bus devices or they can be generic R/W registers.

Writing or reading a scratchpad register does not cause an interrupt to be asserted. Doorbell interrupts can be used for this purpose.They are read/write scratchpad registers.

Error Handling 12

This chapter presents the theory of operation information about the 21555 Error handling capability. See Chapter 16 for specific information about the Error registers.

12.1 Error Signals

This section describes both the primary and secondary PCI bus error signals.

12.1.1 Primary PCI Bus Error Signals

Table 27 describes the primary PCI bus error signals.

Table 27. Primary PCI Bus Error Signals

Signal Name	Type	Description
p_perr_l	STS	Primary PCI interface PERR#. Signal **p_perr_l** is asserted when a data parity error is detected for data received on the primary interface. The timing of **p_perr_l** corresponds to **p_par** driven one clock cycle earlier, and **p_ad** and **p_cbe_l** driven two clock cycles earlier. Signal **p_perr_l** is asserted by the target during write transactions, and by the initiator during read transactions. Upon completion of a transaction, **p_perr_l** is driven to a deasserted state for one clock cycle and is then sustained by an external pull-up resistor.
p_serr_l	OD	Primary PCI interface **SERR#**. Signal **p_serr_l** can be driven low by any device on the primary bus to indicate a system error condition. The 21555 can conditionally assert **p_serr_l** for the following reasons: • Primary bus address parity error. • Downstream posted write data parity error on secondary bus. • Master abort during downstream posted write transaction. • Target abort during downstream posted write transaction. • Downstream posted write transaction discarded. • Downstream delayed write request discarded. • Downstream delayed read request discarded. • Downstream delayed transaction master timeout. • Secondary bus **s_serr_l** assertion. Signal **p_serr_l** is pulled up through an external resistor.

12.1.2 Secondary PCI Bus Error Signals

Table 28 describes the secondary PCI bus error signals.

Table 28. Secondary PCI Bus Arbitration Signals

Signal Name	Type	Description
s_perr_l	STS	Secondary PCI interface **PERR#**. Signal **s_perr_l** is asserted when a data parity error is detected for data received on the secondary interface. The timing of **s_perr_l** corresponds to s_par driven one clock cycle earlier, and **s_ad** driven two clock cycles earlier. Signal **s_perr_l** is asserted by the target during write transactions, and by the initiator during read transactions. Upon completion of a transaction, **s_perr_l** is driven to a deasserted state for one clock cycle and is then sustained by an external pull-up resistor.
s_serr_l	OD	Secondary PCI interface **SERR#**. Signal **s_serr_l** can be driven low by any device on the secondary bus to indicate a system error condition. The 21555 also samples **s_serr_l** as an input and conditionally forwards it to the primary bus on **p_serr_l**. The 21555 can conditionally assert **s_serr_l** for the following reasons: • Secondary bus address parity error. • Upstream posted write data parity error on primary bus. • Master abort during upstream posted write transaction. • Target abort during upstream posted write transaction. • Upstream posted write transaction discarded. • Upstream delayed write request discarded. • Upstream delayed read request discarded. • Upstream delayed transaction master timeout. Signal **s_serr_l** is pulled up through an external resistor.

12.2 Parity Errors

The 21555 checks, forwards, and generates parity on both the primary and secondary buses. When forwarding transactions, the 21555 forwards the data parity condition as queued, whether it is bad parity or good parity. Table 29 describes the 21555's responses to parity errors.

Table 29. Parity Error Responses (Sheet 1 of 3)

Type of Error	Type of Transaction	PER† P|S	Action Taken
Address Parity Error	Primary Bus Transaction	0 | —	• Responds normally to transaction. • Sets primary Detected Parity Error bit. • Forwards transaction with correct parity.
		1 | —	• Does not respond to transaction. • Asserts **p_serr_l** (when enabled). • Sets primary Detected Parity Error bit.
	Secondary Bus Transaction	— | 0	• Responds normally to transaction. • Sets secondary Detected Parity Error bit. • Forwards transaction with correct parity.
		— | 1	• Does not respond to transaction. • Asserts **s_serr_l** (when enabled). • Sets secondary Detected Parity Error bit.
Data Parity Error on Primary Bus	Downstream Posted Write	0 | —	• Forwards transaction with parity error. • Sets primary Detected Parity Error bit.
		1 | —	• Forwards transaction with parity error. • Sets primary Detected Parity Error bit. • Asserts **p_perr_l.**
	Upstream Posted Write	0 | —	Trans action completes normally on primary bus.
		1 | —	• Transaction completes on primary bus. • Sets primary Data Parity Detected bit when **p_perr_l** is asserted.
		1 | 1	• Transaction completes on primary bus. • Sets primary Data Parity Detected bit when **p_perr_l** is asserted. • Asserts **s_serr_l** when no parity error detected on secondary bus.

† PER: Parity Error Response bit (Primary | Secondary).

Table 29. Parity Error Responses (Sheet 2 of 3)

Type of Error	Type of Transaction	PER† P|S	Action Taken
Data Parity Error on Primary Bus	Downstream Delayed Write	0 | —	• Queues and forwards transaction with parity error. • Sets primary Parity Error Detected bit.
		1 | —	• Returns **TRDY#** (and **STOP#** when multiple data phases requested). • Transaction not forwarded. • Sets primary Parity Error Detected bit. • Asserts **p_perr_l.**
	Upstream Delayed Write	0 | —	• Transaction completes normally on primary bus.
		1 | —	• Transaction completes normally on primary bus. • Sets primary Data Parity Detected bit when **p_perr_l** is asserted.
		1 | 1	• Transaction completes normally on primary bus. • Sets primary Data Parity Detected bit when **p_perr_l** is asserted. • Asserts **s_perr_l** when returning **s_trdy_l** to initiator on secondary bus (for both CSR and BAR forwarding mechanisms).
	Downstream Delayed Read	— | —	The 21555 is returning data, all action is taken by initiator.
	Upstream Delayed Read	0 | —	• Returns read data with bad parity to initiator (for both CSR and BAR forwarding mechanisms). • Sets primary Parity Error Detected bit.
		1 | —	• Returns read data with bad parity to initiator (for both CSR and BAR forwarding mechanisms). • Sets primary Parity Error Detected bit. • Sets primary Data Parity Detected bit. • Asserts **p_perr_l.**
	Configuration Register or CSR Write	0 | —	• Writes the data normally. • Sets the primary Parity Error Detected bit.
		1 | —	• Writes the data normally. • Sets the primary Parity Error Detected bit. • Asserts **p_perr_l**.
	Configuration Register or CSR Read	— | —	Returns read data normally.

† PER: Parity Error Response bit (Primary | Secondary).

Table 29. Parity Error Responses (Sheet 3 of 3)

Type of Error	Type of Transaction	PER† P|S	Action Taken
Data Parity Error on Secondary Bus	Downstream Posted Write	— | 0	Transaction completes normally on secondary bus.
		— | 1	• Transaction completes on secondary bus. • Sets secondary Data Parity Detected bit when **s_perr_l** is asserted.
		1 | 1	• Transaction completes on secondary bus. • Sets secondary Data Parity Detected bit when **s_perr_l** is asserted. • Asserts **p_serr_l** when no parity error detected on primary bus.
	Upstream Posted Write	— | 0	• Forwards transaction with parity error. • Sets secondary Detected Parity Error bit.
		— | 1	• Forwards transaction with parity error. • Sets secondary Detected Parity Error bit. • Asserts **s_perr_l.**
	Downstream Delayed Write	— | 0	Transaction completes normally on secondary bus.
		— | 1	• Transaction completes normally on secondary bus. • Sets secondary Data Parity Detected bit when **s_perr_l** is asserted.
		1 | 1	• Transaction completes normally on secondary bus. • Sets secondary Data Parity Detected bit when **s_perr_l** is asserted. • Asserts **p_perr_l** when returning **p_trdy_l** to initiator on primary bus (for both CSR and BAR forwarding mechanisms).
	Upstream Delayed Write	— | 0	Transaction completes normally on secondary bus.
		— | 1	• Returns **TRDY#** (and **STOP#** if multiple data phases requested). • Transaction not forwarded. • Sets secondary Parity Error Detected bit. • Asserts **s_perr_l.**
	Downstream Delayed Read	— | 0	• Returns read data with bad parity to initiator (for both CSR and BAR forwarding mechanisms). • Sets secondary Parity Error Detected bit.
		— | 1	• Returns read data with bad parity to initiator (for both CSR and BAR forwarding mechanisms). • Sets secondary Parity Error Detected bit. • Sets secondary Data Parity Detected bit. • Asserts **s_perr_l.**
	Upstream Delayed Read	— | —	The 21555 is returning data, all action is taken by initiator.
	Configuration Register or CSR Write	— | 0	• Writes the data normally. • Sets the secondary Parity Error Detected bit.
		— | 1	• Writes the data normally. • Sets the secondary Parity Error Detected bit. • Asserts **s_perr_l.**
	Configuration Register or CSR Read	— | —	Returns read data normally.

† PER: Parity Error Response bit (Primary | Secondary).

12.3 System Error (SERR#) Reporting

The 21555 has two system error pins. Signal **p_serr_l** reports system errors on the primary interface, and **s_serr_l** reports system errors on the secondary interface. For the 21555 to assert the **SERR**# signal for that interface, the **SERR#** Enable must be set in the Command Configuration register corresponding to that interface. In addition, each device-specific condition has a disable bit for each interface. When a disable bit for a particular condition is set, **SERR#** assertion is masked for that condition. **SERR#** may be asserted for any of the following conditions:

- Address parity error (disabled by the corresponding Parity Error Enable bit).
- Parity error reported on target bus only during a posted write transaction (disabled by the corresponding Parity Error Enable bit).
- Target abort detected during posted write transaction.

  Note: The Master Abort Mode Bit in the Chip Control register must be set in order for **SERR** to assert on the following condition.

- Master abort detected during posted write transaction.
- Posted write discarded after 2^{24} target retries received from target.
- Delayed write request discarded after 2^{24} target retries received from target.
- Delayed read request discarded after 2^{24} target retries received from target.
- Delayed transaction completion discarded after master timeout counter expired.

For the above conditions, **SERR#** is asserted on the interface corresponding to the location of the initiator of the transaction.

When the **SERR#** Forward Enable bit in the Chip Control 0 configuration register is set, the 21555 asserts **p_serr_l** when it detects **s_serr_l** asserted, including those cases where the 21555 has asserted **s_serr_l**.

JTAG Test Port 13

This chapter presents the theory of operation information about the 21555 JTAG interface. See Chapter 16 for specific information about the JTAG registers.

The 21555's implementation of the JTAG test port is according to IEEE Std. 1149.1, IEEE Standard Test Access Port and Boundary-Scan Architecture.

The JTAG test port consists of the following:

- A 5-signal test port interface.
- A test access port controller.
- An instruction register.
- A bypass register.
- A boundary-scan register.

13.1 JTAG Signals

The JTAG test access port is to be used only while the 21555 is not operating. Table 30 describes JTAG signals.

Table 30. JTAG Signals

Signal Name	Type	Description
tck	I	JTAG boundary-scan clock. Signal **tck** is the JTAG logic control clock. This pin has an internal weak pull-down resistor.
tdi	I	JTAG serial data in. Signal **tdi** is the serial input through which JTAG instructions and test data enter the JTAG interface. The new data on **tdi** is sampled on the rising edge of **tck**. An unterminated **tdi** is pulled high by a weak pull-up resistor internal to the device.
tdo	O	JTAG serial data out. Signal **tdo** is the serial output through which test instructions and data from the test logic leave the 21555.
tms	I	The JTAG test mode select pin, **tms** causes state transitions in the Test Access Port (TAP) controller. The **tms** signal is pulled high by a weak pull-up resistor internal to the device. If this pin is low while **t_rst_l** is low the device can enter an unsupported mode. Other devices that are not on early power and are connected to the JTAG Scan Chain, pull **tms** low during Hot Insertion causing the 21555 to enter the unsupported mode. During the Hot Insertion isolate this signal from other JTAG devices on the circuit board or JTAG scan chain.
trst_l	I	JTAG TAP reset and disable. When low, JTAG is disabled and the TAP controller is asynchronously forced into the reset state, which in turn asynchronously initializes other test logic. An unterminated **trst_l** is pulled high by a weak pull-up resistor internal to the device. The TAP controller must be reset before the JTAG circuits can function. For normal JTAG TAP port operation, this signal must be high. Prior to normal 21555 operation, this signal must be strobed low or pulled low with a 1kΩ resistor.

13.2 Test Access Port Controller

The test access port controller is a finite-state machine that interprets IEEE 1149.1 protocols received through the **tms** signal. The state transitions in the controller are caused by the **tms** signal on the rising edge of **tck**. In each state, the controller generates appropriate clock and control signals that control the operation of the test features. After entry into a state, test feature operations are initiated on the rising edge of **tck**.

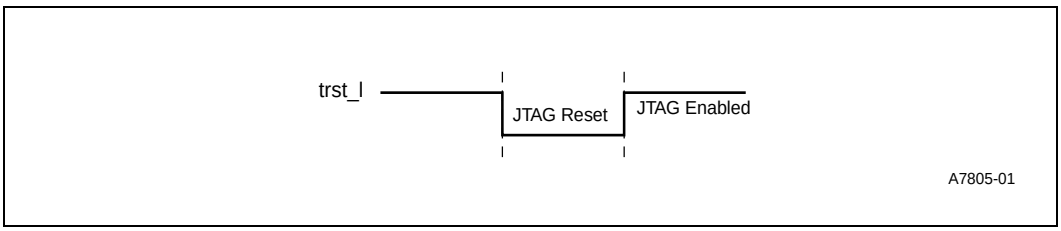
13.2.1 Initialization

The test access port controller and the instruction register output latches are initialized and JTAG is disabled while the **trst_l** input is asserted low (see Figure 26). While signal **trst_l** is low, the test access port controller enters the test-logic reset state. This results in the instruction register being reset which holds the bypass register instruction. During test-logic reset state, all JTAG test logic is disabled, and the device performs normal functions. The test access port controller leaves this state only after **trst_l** (low) goes high and an appropriate JTAG test operation sequence is sent on the tms and tck pins.

For the 21555 to operate properly, the JTAG logic must be reset. The controller will reset:

- Asynchronously with the assertion of **trst_l** .
- Synchronously after five **tck** clock cycles, with **tms** held high.

Figure 26. Signal trst_l States



Note: Prior to normal 21555 operation, this signal must be strobed low or pulled low with a 1kΩ resistor.

I2O Support 14

This chapter presents the theory of operation information about the 21555 I20 support. See Chapter 16 for specific information about I20 registers.

The 21555 implements an I2O messaging unit to allow passing of I2O messages between the host system and the local subsystem which is called IOP in I2O nomenclature.

For the host system to identify the local subsystem as an I2O compliant IOP, the class code must be preloaded to indicate I2O support:

- The Base Class is loaded with the code for intelligent I/O controllers (0Eh).
- The Sub-class Code is loaded with the code indicating I2O conformance (00h).
- The Programming Interface is loaded with (01h) to indicate 32-bit data width, 32-bit addressing, little endian, with support of the outbound post status and mask registers.
- The I2O Enable bit in the Chip Control 1 configuration register must be set to a 1 to enable the I2O message unit.

Otherwise, accesses to the I2O Inbound Queue and I2O Outbound Queue result in TRDY# and a discard of data for memory writes and TRDY#, and return of FFFFFFFFh for memory reads.

The 21555 implements two predefined I2O registers in CSR space that allow access to the I2O Inbound Queue, at offset 40h, and the I2O Outbound Queue, at offset 44h. The actual queues are located in local memory. Each queue has a Post_List FIFO and a Free_List FIFO.

- The Post_List contains I2O Message Frame Addresses (MFAs).
- The Free_List contains empty MFAs.

The 21555 implements hardware to control the FIFOs from the host side. Control of the FIFOs from the local processor side is done in software.

14.1 Inbound Message Passing

An inbound message is passed from the host processor to the local processor in the following steps:

1. The host processor removes an empty MFA, if available, from the head of the Inbound Free_List.
2. The host processor posts an MFA containing the address of the message frame to the tail of the Inbound Post_List.
3. The I2O controller interrupts the local processor, indicating that an MFA exists in the Inbound Post_List.
4. The local processor retrieves the MFA from the head of the Inbound Post_List.
5. After the local processor consumes the message, it replaces the empty MFA onto the tail of the Inbound Free_List.

The 21555 implements the following hardware for the Inbound Queue:

- Table 85, “I2O Inbound Queue” on page 166 register at CSR offset 40h.
- Table 87, “I2O Inbound Free_List Head Pointer” on page 167 at CSR offset 48h.
- Table 88, “I2O Inbound Post_List Tail Pointer” on page 167 at CSR offset 4Ch.
- Table 91, “I2O Inbound Post_List Counter” on page 168 at CSR offset 58h.
- Table 92, “I2O Inbound Free_List Counter” on page 168 at CSR offset 5Ch.
- Table 83, “I2O Inbound Post_List Status” on page 165 at CSR offset 38h.
- Table 84, “I2O Inbound Post_List Interrupt Mask” on page 166 at CSR offset 3Ch.

When the host processor has a message to pass to the local processor, it first reads the Inbound Queue location at 40h to remove an empty MFA from the Inbound Free_List. The 21555 maintains an on silicon 2 Dword buffer to hold the next two empty MFAs from the Inbound Free_List. When this buffer is not empty, the 21555 returns **TRDY#** and the next empty MFA from its buffer. When the internal buffer empties as a result of this read operation, and if the Inbound Free_List Counter is non-zero, the 21555 automatically reads one or two more MFAs from the Inbound Free_List as described in the following paragraph.

When the 2 Dword buffer is empty, the 21555 treats a read to location 44h as a delayed memory read transaction. The address that the 21555 uses to initiate the transaction on the secondary bus is the current value of the Outbound Post_List Head Pointer. When the Outbound Post_List Counter is non-zero, the 21555 places the read request in a downstream delayed transaction queue entry reserved for fetching outbound MFAs as follows:

- When the Outbound Post_List Counter is 2 or higher, the 21555 performs a 2 Dword secondary bus memory read starting at the location addressed by the Outbound Post_List head pointer and places the read data in the 2 Dword buffer.
- When the Outbound Post_List Counter is 1, the 21555 performs a single Dword read.

When the read completes on the secondary bus, the 21555 decrements the Outbound Post_List Counter by 1 or 2, respectively. The 21555 also increments the Outbound Post_List Head Pointer by either 1 or 2 Dwords, respectively. When the initiator repeats the read of CSR location 44h, the 21555 returns the next MFA to the host.

When the Inbound Free_List Counter is zero and the prefetch buffer is empty, the 21555 immediately returns FFFFFFFFh to the host and does not enter the transaction in the delayed transaction queue. The 21555 also does not decrement the Inbound Free_List Counter or increment the Inbound Free_List Head pointer.

Once the host obtains an empty MFA from the Inbound Free_List, it may then post a message to the local processor. The host processor posts a message to the Inbound Queue by writing the MFA to offset 40h. The 21555 treats the write to location 40h as a posted write; that is, it returns **TRDY#** to the initiator and places the write data in the posted write queue. The 21555 translates the address to the current value of the Inbound Post_List Tail Pointer. Once the MFA is queued in the posted write buffer, the 21555 increments the Inbound Post_List Tail Pointer by 1. The 21555 can continue to accept posted inbound MFAs as long as there is room in the downstream posted write queue. The 21555 performs a secondary bus memory write of this data to the location addressed by the Inbound Post_List Tail Pointer. When this write is completed on the secondary bus, the 21555 increments the Inbound Post_List counter by 1. As long as the Inbound Post_List counter is non-zero, the 21555 sets the Inbound Post_List status to 1, and asserts **s_inta_l** if the Inbound Post_List Mask bit is zero. Signal **s_inta_l** remains asserted until either the Inbound Post_List counter is zero or the Inbound Post_List Mask bit is set.

The local processor manages the removal of the MFA from the Inbound Post_List and the replacement of the empty MFA to the Inbound Free_List in software. The 21555 does not implement inbound queue pointers that would be used by the local processor. However, the local processor must manage the 21555’s Inbound Post_List counter when it removes an MFA. The Inbound Free_List Counter is needed so that the 21555 can determine when the Inbound Free_List is empty and must return FFFFFFFFh to the host processor instead of an empty MFA. When the local

processor removes the message from the Inbound Post_List, it must write bit 31 of the Inbound Post_List counter with a 0, which causes the 21555 to decrement the Inbound Post_List counter by 1. When the counter decrements to zero, the 21555 deasserts **s_inta_l**, indicating that there are no more posted MFAs in the Inbound Queue.

Once the local processor consumes the inbound message from the host, it replaces the empty MFA onto the end of the Inbound Free_List. Again, the 21555 does not manage the pointers for this operation, so the local processor must manage this in software. However, the local processor manages the 21555's Inbound Free_List Counter when it replaces an empty MFA. When the local processor replaces the empty MFA to the Inbound Free_List, it must write bit 31 of the Inbound Free_List Counter with a zero, which causes the 21555 to increment the Inbound Free_List Counter by 1.

14.2 Outbound Message Passing

An outbound message is passed from the local processor to the host processor in the following steps:

1. The local processor removes an empty MFA, if available, from the head of the Outbound Free_List.
2. The local processor posts an MFA containing the address of the message frame to the tail of the Outbound Post_List.
3. The I2O Controller interrupts the host processor, indicating that an MFA exists in the Outbound Post_List.
4. The host processor retrieves the MFA from the head of the Outbound Post_List.
5. After the host processor consumes the message, it replaces the empty MFA onto the tail of the Outbound Free_List.

The 21555 implements the following hardware for the Outbound Queue:

- Table 86, "I2O Outbound Queue" on page 166 register at CSR offset 44h.
- Table 89, "I2O Outbound Free_List Tail Pointer" on page 167 at CSR offset 50h.
- Table 90, "I2O Outbound Post_List Head Pointer" on page 167 at CSR offset 54h.
- Table 93, "I2O Outbound Post_List Counter" on page 169 at CSR offset 60h.
- Table 94, "I2O Outbound Free_List Counter" on page 169 at CSR offset 64h.
- Table 81, "I2O Outbound Post_List Status" on page 165at CSR offset 30h.
- Table 82, "I2O Outbound Post_List Interrupt Mask" on page 165 at CSR offset 34h.

When the local processor has a message to pass to the host processor, it must remove an empty MFA from the head of the Outbound Free_List. The 21555 does not implement outbound queue pointers that are used by the local processor. However, the local processor manages the 21555's Outbound Free_List counter when it removes an empty MFA. When the local processor removes the empty MFA from the Outbound Free_List, it must write bit 31of the Outbound Free_List counter to a zero, causing the 21555 to decrement the Outbound Free_List counter by 1. The 21555 does not use the value of this counter internally, but makes this function available to track the number of empty MFAs in the Outbound Free_List

When the local processor posts a message to the Outbound Post_List, it must write bit 31 of the Outbound Post_List Counter to a zero, which causes the 21555 to increment the counter by 1. A non-zero value in the Outbound Post_List Counter indicates that the Outbound Post_List contains MFAs intended for the host processor. When the Outbound Post_List Mask bit is zero, upon detection of a non-zero value in the Outbound Post_List counter or if the onchip outbound prefetch buffer is not empty, the 21555 sets the Outbound Post_List status to 1

and asserts **p_inta_l** to indicate to the host processor that one or more MFAs exist in the Outbound Post_List . Signal **p_inta_l** remains asserted until either the Outbound Post_List Counter is zero and the outbound prefetch buffer empties, or the Outbound Post_List Mask bit is set.

The host processor removes the message from the Outbound Post_List by reading the 21555 CSR offset 44h. The 21555 maintains a 2 Dword outbound prefetch buffer to hold the next two MFAs from the Outbound Post_List. When this buffer is not empty and the Outbound Post_List Counter is non-zero, the 21555 returns **TRDY#** and the next MFA from its buffer. When the internal buffer empties as a result of this read operation, the 21555 automatically reads one or two more MFAs from the Outbound Post_List as described in the following paragraph.

When the 2 Dword buffer is empty, the 21555 treats a read to location 44h as a delayed memory read transaction. The address that the 21555 uses to initiate the transaction on the secondary bus is the current value of the Outbound Post_List Head Pointer. When the Outbound Post_List Counter is non-zero, the 21555 places the read request in a downstream delayed transaction queue entry reserved for fetching outbound MFAs as follows:

- When the Outbound Post_List Counter is 2 or higher, the 21555 performs a 2 Dword secondary bus memory read starting at the location addressed by the Outbound Post_List head pointer and places the read data in the 2 Dword buffer.
- When the Outbound Post_List Counter is 1, the 21555 performs a single Dword read.

When the read completes on the secondary bus, the 21555 decrements the Outbound Post_List Counter by 1 or 2, respectively. The 21555 increments the Outbound Post_List Head Pointer by either 1 or 2 Dwords, respectively, as well. When the initiator repeats the read of CSR location 44h, the 21555 returns the next MFA to the host.

When the Outbound Post_List Counter is zero and the prefetch buffer is empty, the 21555 immediately returns FFFFFFFFh to the host and does not enter the transaction in the delayed transaction queue and also does not decrement the Outbound Post_List Counter. When the counter decrements to zero and the 2 Dword prefetch buffer is empty, the 21555 deasserts **p_inta_l**, indicating that there are no more posted MFAs in the Outbound Queue.

Once the host processor consumes the outbound message from the local processor, it replaces the empty MFA onto the end of the Outbound Free_List. When the host processor replaces the empty MFA to the Outbound Free_List, it writes the Outbound Queue at the 21555 CSR offset 44h. The 21555 treats the write to location 44h as a posted write; that is, it returns **TRDY#** to the initiator and places the write data in the downstream posted write queue. The 21555 translates the address to the current value of the Outbound Free_List tail pointer. Once the empty MFA is queued in the posted write buffer, the 21555 increments the Outbound Free_List tail pointer. The 21555 can continue to accept writes to this address as long as there is room in the downstream posted write queue. The 21555 writes the data to the secondary bus location addressed by the Outbound Free_List tail pointer. When the write is completed, the 21555 increments the Outbound Free_List counter.

14.3 Notes

- Read transactions to I2O Inbound and Outbound Queues at 40h and 44h are not ordered with respect to transactions in the posted write or delayed transaction queues. Reads to these two registers will not flush posted writes. Write transactions to these registers are placed in the posted write queue, and follow the same ordering rules as other posted memory writes.
- When the 21555 detects parity errors, a master abort or target abort during a read or write access to the Inbound or Outbound Queues, the 21555 treats the error condition the same way as it does any other delayed read or posted write.
- The 21555 queue pointers support FIFO sizes of 256, 512, 1K, 2K, 4K, 8K, 16K, and 32K entries. The number of entries is selectable in the Table 78, “Chip Control 1 Register” on page 160. The wrap function for all of the I2O pointers maintained by the 21555 is performed in hardware; therefore, all FIFOs must be located in an aligned address boundary.

- All MFA counters maintained by the 21555 may be individually loaded with any data value by writing a 1 to bit 31 of the corresponding counter Dword offset. When either the Inbound Free_List Counter or the Outbound Post_List Counter is loaded, the 21555 discards any prefetched data in the corresponding prefetch buffer.
- The 21555 actions are unpredictable in response to primary bus transactions addressing the I2O Outbound or Inbound Queues while a corresponding counter load is occurring on the secondary bus. The counters will load and increment even if the I2O Enable is not set.
- The 21555 I2O counters are consistent with the number of entries in the various lists from the secondary bus, or local memory, point of view. This means that counters do not include any MFAs that exist in the 21555 posting or prefetch buffers. The counters include only those entries that are present in local memory. The Outbound Free_List and Inbound Post_List pointers are consistent with the primary bus viewpoint. This means that the value of the head and tail pointers that the 21555 implements includes any data in the 21555 posted write buffers as part of the lists. The pointers corresponding to the Outbound Post_List and Inbound Free_List should not be considered to be consistent with the number of entries in these lists.
- The 21555 I2O counters are consistent with the number of entries in the various lists from the secondary bus, or local memory, point of view. This means that counters do not include any MFAs that exist in the 21555 posting or prefetch buffers. The counters include only those entries that are present in local memory.
- When the I2O Enable bit is disabled after the I2O message unit is operational, additional reads to 40h and 44h returns data that remains in the corresponding prefetch buffer until the prefetch buffer is emptied. After the prefetch buffer is emptied, the 21555 returns FFFF FFFFh.
- When the secondary interface Master Enable bit is disabled:
  - — Reads to 40h and 44h are FFFF FFFFh.
  - — Writes to 40h and 44h are queued in the downstream posted write queue, but not delivered until the master enable bit is set.

VPD Support 15

This chapter presents the theory of operation information about the 21555 Vital Product Data (VPD) support. See Chapter 16 for specific information about the VPD registers.

The 21555 provides VPD support through its serial ROM interface. Note that VPD support in the Expansion ROM as described in the *PCI Local Bus Specification, Revision 2.2*, is transparent to the 21555 and is not described in this document.

VPD is stored in the last 3K bits (384 bytes) of the serial ROM. The first 1K bits (128 bytes) of the VPD space are designated as read only and cannot be written from the VPD serial ROM register interface. The upper 2K bits (256 bytes) are designated as read/write from the VPD serial ROM register interface. Only VPD data can be accessed through this interface. To read or write any serial ROM location, the CSR serial ROM access mechanism should be used, as described in Chapter 9 .

15.1 Reading VPD Information

A read can occur to any location in VPD space. Valid VPD byte addresses are 17F:000h. To read VPD information from the serial ROM, the following steps must be taken:

1. The VPD address and VPD Flag bits are written. This requires a write to bytes E7:E6h, where the low 9 bits carry the VPD byte address and bit 15 is a 0 (zero), indicating a read operation. The 21555 adds the VPD base address, 080h, to the VPD byte address to obtain the serial ROM address and perform a read of 4 bytes. The VPD address has no alignment requirements; it can start on any byte boundary.
2. The VPD Flag bit is polled. When the 21555 returns a 1, the read is complete.
3. The VPD information can be read from the VPD Data register at bytes EB:E8. Byte 0 contains the data at the location referenced by the VPD Address register. Bytes 3:1 contain successive bytes.

The 21555 always performs a 4-byte read, regardless of the VPD byte address. Therefore, when the VPD byte address is one of the last 3 bytes in VPD space, the serial ROM address wraps. The remaining 1, 2, or 3 bytes contain invalid data and should be ignored.

The VPD Address and VPD Data registers should not be written while a VPD read operation is taking place, otherwise results are unpredictable.

15.2 Writing VPD Information

A write can occur only to the last 2 Kb (256 bytes) of VPD Space. Valid VPD byte addresses for write operations are 17F:080h. To write VPD information from the serial ROM, the following steps must be taken:

1. The VPD data register is written with 4 bytes of data. Byte 0 contains the data to be written to the location referenced by the VPD byte address. The value in bytes 3:1 of the VPD Data register is written to successive byte locations in VPD space.
2. The VPD address and VPD Flag bits are written. This requires a write to bytes E7:E6h, where the low 9 bits carry the VPD byte address and bit 15 is a 1, indicating a write operation. The 21555 adds the VPD base address, 080h, to the VPD byte address to obtain the serial ROM address and perform a write of 4 bytes. The VPD address has no alignment requirements; it can start on any byte boundary.
3. The VPD Flag bit is polled. When the 21555 returns a 0, the write is complete.

When a write is attempted to a location in the first 1Kb of serial ROM space (address bits 8:7 is 00b), the 21555 does not perform the write operation and clears the flag bit immediately. When the VPD byte address is one of the last three byte locations in VPD space, the 21555 only completes those writes to the end of VPD space, that is, it performs either a 3-, 2-, or 1-byte write.

The 21555 tracks whether the serial ROM is enabled for writes. If the serial ROM is write disabled when a VPD write is attempted, the 21555 first automatically performs a write enable operation to the serial ROM. The 21555 does not automatically perform this operation for serial writes using the CSR mechanism.

The VPD Address and VPD Data registers should not be written while any other VPD, serial ROM, or parallel ROM (PROM) operation is taking place, otherwise results are unpredictable.

List of Registers 16

This chapter contains reference information about all of the 21555 registers. Table 31 is a cross reference between the sections in this chapter to there accompanying theory of operation chapters.

Table 31. Register Cross Reference Table

Theory of Operation Chapter	Register Reference Information.
Chapter 4, “Address Decoding”	Section 16.4, “Address Decoding” on page 130
Chapter 3, “Signal Descriptions” Chapter 5, “PCI Bus Transactions”	Section 16.5, “PCI Registers” on page 147
Chapter 6, “Initialization Requirements”	Section 16.13, “Init Registers” on page 185
Chapter 11, “Interrupt and Scratchpad Registers”	Section 16.7, “Interrupt Registers” on page 170 Section 16.8, “Scratchpad Registers” on page 174
Chapter 8, “Parallel ROM Interface”	Section 16.9, “PROM Registers” on page 175
Chapter 9, “Serial ROM Interface”	Section 16.10, “SROM Registers” on page 179
Chapter 10, “Arbitration”	Section 16.11, “Arbiter Control” on page 183
Chapter 12, “Error Handling”	Section 16.12, “Error Registers” on page 183
Chapter 13, “JTAG Test Port”	Section 16.14, “JTAG Registers” on page 190
Chapter 14, “I2O Support”	Section 16.6, “I2O Registers” on page 165
Chapter 15, “VPD Support”	Section 16.15, “VPD Registers” on page 192

16.1 Register Summary

This chapter lists the 21555 configuration space registers and the CSR address map registers. A description of the notes used in the tables used in this chapter are listed as follows:

- Byte offsets that are specific to the primary or secondary interfaces are followed by a (P) or (S) respectively. All other byte offsets refer to both the primary and secondary address spaces. The configuration registers are listed in order of primary byte offset.
- For read and write access, the following apply:
  - **Y**: accessible from both primary and secondary interface.
  - **N**: not accessible from either interface.
  - **Primary**: for writes, only write from primary interface; for reads, reads as 0 from secondary interface.
  - **Secondary**: for writes, only write from secondary interface; for reads, reads as 0 from primary interface.
  - Special cases (for example, W1TC, W1TS, and R0TS) are noted.
- Some registers contain fields or bits with different access types (for example, R, R/W, W1TC) which are not detailed in this table. See the individual register description for detailed information.
- Not all bits in every register denoted as preloadable are necessarily preloaded. See the individual register description for detailed information.

- "Via Setup" refers to the base address setup register corresponding to that BAR

16.2 Configuration Registers

Table 32 lists the configuration space address registers.

Table 32. Configuration Space Address Register (Sheet 1 of 5)

Byte Offset (Hex)	Register Name	Reset Value (Hex)	Preload	Write Access	Read Access
01:00 41:40	Vendor ID Register, page 148	1011	—	N	Y
03:02 43:42	Device ID Register, page 148	B555	—	N	Y
05:04 (P) 45:44 (S)	Primary and Secondary Command Registers, page 149	0000	—	Y	Y
07:06 (P) 47:46 (S)	Primary and Secondary Status Registers, page 150	0290	—	Y	Y
08 48	Revision ID (Rev ID) Register, page 151	Device dependent	—	N	Y
0B:09 (P) 4B:49 (S)	Primary and Secondary Class Code Registers, page 152	068000	Y	Secondary	Y
0C (P) 4C (S)	Primary and Secondary Cache Line Size Registers, page 152	00	—	Y	Y
0D (P) 4D (S)	Primary Latency and Secondary Master Latency Timer Registers, page 153	00	—	Y	Y
0E 4E	Header Type Register, page 153	00	—	N	Y
0F 4F	BiST Register, page 153	00	Y	[6] Y [7,3:0] Secondary	Y
13:10 (P) 53:50 (S)	Primary CSR and Memory 0 BAR	00000000	Via Setup	Y	Y
17:14 (P) 57:54 (S)	Primary CSR I/O BAR	00000001	—	Y	Y
1B:18 (P) 5B:58 (S)	Downstream Memory 1 BAR	000000000	Via Setup	Via Setup	Y
1F:1C (P) 5F:5C (S)	Downstream Memory 2 BAR	000000000	Via Setup	Via Setup	Y
23:20 (P) 63:60 (S)	Downstream Memory 3 BAR	00000000	Via Setup	Via Setup	Y
27:24 (P) 67:64 (S)	Downstream Memory 3 Upper 32 Bits	00000000	Via Setup	Via Setup	Y
2B:28 6B:67	Reserved	00000000	—	N	Y
2D:2C 6D:6C	Subsystem Vendor ID Register, page 154D	0000	Y	Secondary	Y

Table 32. Configuration Space Address Register (Sheet 2 of 5)

Byte Offset (Hex)	Register Name	Reset Value (Hex)	Preload	Write Access	Read Access
2F:2E 6F:6E	Subsystem ID Register, page 154	0000	Y	Secondary	Y
33:30 (P) 73:70 (S)	Primary Expansion ROM BAR, page 175	00000000	Via Setup	Via Setup	Y
34 74	Enhanced Capabilities Pointer Register, page 154	DC	—	N	Y
37:35 (P) 77:75 (S)	Reserved	000000	—	N	Y
3B:38 (P) 7B:78 (S)	Reserved	00000000	—	N	Y
3C (P) 7C (S)	Primary and Secondary Interrupt Line Registers, page 154	00	—	Y	Y
3D (P) 7D (S)	Primary and Secondary Interrupt Pin Registers, page 155	01	—	N	Y
3E (P) 7E (S)	Primary and Secondary Minimum Grant Registers, page 155	00	Y	Secondary	Y
3F (P) 7F (S)	Primary and Secondary Maximum Latency Registers, page 155	00	Y	Secondary	Y
45:44 (P) 05:04 (S)	Primary and Secondary Command Registers, page 149	0000			
47:46 (P) 07:06 (S)	Primary and Secondary Status Registers, page 150	0290	—	Y	Y
4B:49 (P) 0B:09 (S)	Primary and Secondary Class Code Registers, page 152	068000	Y	N	Y
4C (P) 0C (S)	Primary and Secondary Cache Line Size Registers, page 152	00	—	Y	Y
4D (P) 0D (S)	Primary Latency and Secondary Master Latency Timer Registers, page 153	00	—	Y	Y
53:50 (p) 13:10 (s)	Secondary CSR Memory BAR	00000000	—	Y	Y
57:54 (p) 17:14 (s)	Secondary CSR I/O BAR	00000001	—	Y	Y
5B:58 (P) 1B:18 (S)	Upstream I/O or Memory 0 BAR	00000000	Via Setup	Via Setup	Y
5F:5C (P) 1F:1C (S)	Upstream Memory 1 BAR	00000000	Via Setup	Via Setup	Y
63:60 (P) 23:20 (S)	Upstream Memory 2 BAR	00000000	Via Chip Control 1	Via Chip Control 1	Y
67:64 (P) 27:24 (S)	Reserved	00000000	—	N	Y
73:70 (P) 33:30 (S)	Reserved	00000000	—	N	Y
7C(P) 3C (S)	Primary and Secondary Interrupt Line Registers, page 154	00	—	Y	Y

Table 32. Configuration Space Address Register (Sheet 3 of 5)

Byte Offset (Hex)	Register Name	Reset Value (Hex)	Preload	Write Access	Read Access
7D (P) 3D (S)	Primary and Secondary Interrupt Pin Registers, page 155	00	—	N	Y
7E (P) 3E (S)	Primary and Secondary Minimum Grant Registers, page 155	00	Y	N	Y
7F (P) 3F (S)	Primary and Secondary Maximum Latency Registers, page 155	00	Y	N	Y
83:80	Downstream and Upstream Configuration Address Registers, page 141	Indeterminate	—	Primary	Y
87:84	Downstream Configuration Data and Upstream Configuration Data Registers, page 142	Indeterminate	—	Primary	Primary
8B:88	Downstream and Upstream Configuration Address Registers, page 141	Indeterminate	—	Secondary	Y
8F:8C	Downstream Configuration Data and Upstream Configuration Data Registers, page 142	Indeterminate	—	Secondary	Secondary
90	Configuration Own Bits Register, page 142	00	—	N	Primary Read-0-to- set
91		00	—	N	Secondary Read-0-to- set
92:93	Configuration CSR, page 143	0000	—	Y	Y
9B:98	Downstream I/O or Memory 1 and Upstream I/O or Memory 0 Translated Base Register, page 136	Indeterminate	—	Y	Y
A7:A4		Indeterminate	—	Y	Y
97:94	Downstream Memory 0, 2, 3, and Upstream Memory 1 Translated Base Register, page 137	Indeterminate	—	Y	Y
9F:9C		Indeterminate	—	Y	Y
A3:A0		Indeterminate	—	Y	Y
AB:A8		Indeterminate	—	Y	Y
AF:AC	Downstream Memory 0, 2, 3, and Upstream Memory 1 Setup Registers, page 139	FFFFF000	Y	Secondary	Y
B7:B4		00000000	Y	Secondary	Y
BB:B8	Downstream Memory 2 Setup Downstream Memory 3 Setup	00000000	Y	Secondary	Y
B3:B0	Downstream I/O or Memory 1 and Upstream I/O or Memory 0 Setup Registers, page 138	00000000	Y	Secondary	Y
BF:BC	Upper 32 Bits Downstream Memory 3 Setup Register, page 140	00000000	Y	Secondary	Y
C3:C0	Primary Expansion ROM Setup Register, page 176	00000000	Y	Secondary	Y

Table 32. Configuration Space Address Register (Sheet 4 of 5)

Byte Offset (Hex)	Register Name	Reset Value (Hex)	Preload	Write Access	Read Access
C7:C4	Downstream I/O or Memory 1 and Upstream I/O or Memory 0 Setup Registers, page 138	00000000	Y	Secondary	Y
CB:C8	Downstream Memory 0, 2, 3, and Upstream Memory 1 Setup Registers, page 139	00000000	Y	Secondary	Y
CD:CC	Chip Control 0 Register, page 156	0y00 y = 0 or 4 (Strapped)	Y	[15:11, 9:0]Y [10] secondary	Y
CF:CE	Chip Control 1 Register, page 160	0000	Y	Y	Y
D1:D0	Chip Status Register, page 162	0000	—	W1TC	Y
D3:D2	Arbiter Control Register, page 183	0200	Y	Y	Y
D4	Primary SERR# Disable Register, page 184	00	Y	Y	Y
D5	Secondary SERR# Disable Register, page 184	00	Y	Y	Y
D7:D6	Mode Setting Configuration Register, page 179 Determined by Parallel ROM Strapping Options	0000	—	N	Y
DB:D8	Reset Control Register, page 188	0000	—	Primary	Y
DC	Power Management ECP ID and Next Pointer Register, page 185	01	—	N	Y
DD	Power Management Next Ptr	E4	—	N	N
DF:DE	Power Management Capabilities Register, page 186	0001	Y	Secondary	Y
E1:E0	Power Management Control and Status Register, page 187	0000	Y	Y	Y
E2	PMCSR Bridge Support Extensions, page 187	00	—	N	Y
E3	Power Management Data Register, page 188	00	Y	N	Y
E4	Vital Product Data (VPD) ECP ID and Next Pointer Register, page 192	04	—	N	Y
E5	VPD Cap ID VPD Next Ptr	EC	—	N	N
E7:E6	Vital Product Data (VPD) Address Register, page 193 VPD Address	Indeterminate	—	Y	Y
EB:E8	VPD Data Register, page 193 VPD Data	Indeterminate	—	Y	Y

Table 32. Configuration Space Address Register (Sheet 5 of 5)

Byte Offset (Hex)	Register Name	Reset Value (Hex)	Preload	Write Access	Read Access
EC	CompactPCI Hot-Swap Capability Identifier and Next Pointer Register, page 189 Hot-Swap Cap ID Hot-Swap Next Ptr	06	—	Secondary	N
ED		00	—	N	N
EE	CompactPCI Hot-Swap Control Register, page 189 Hot-Swap Control	00x1000b	—	Y	N
FF:F0	Reserved	00000000	—	N	Y

16.3 Control and Status Registers

The control and status registers are memory mapped in the Primary CSR and Memory 0 Base Address window and the Secondary CSR Base Address window. These registers are I/O mapped in the Primary CSR I/O BAse Address window and the Secondary CSR I/O Base Address window. Offsets are referenced from these base addresses. Table 33 lists the CSR summary. 017:014

Table 33. CSR Address Map (Sheet 1 of 5)

Byte Offset (Hex)	Register Name	Reset Value	Write Access	Read Access
003:000	Downstream and Upstream Configuration Address Registers, page 141 Downstream Configuration Address	Indeterminate	Primary	Y
007:004	Downstream Configuration Data and Upstream Configuration Data Registers, page 142 Downstream Configuration Data	Indeterminate	Primary	Primary
00B:008	Downstream and Upstream Configuration Address Registers, page 141 Upstream Configuration Address	Indeterminate	Secondary	Y
00F:00C	Downstream Configuration Data and Upstream Configuration Data Registers, page 142 Upstream Configuration Data	Indeterminate	Secondary	Secondary
010	Configuration Own Bits Register, page 142 Configuration Own Byte 0	00	N	Primary Read-0-to-set
011	Configuration Own Byte 1	00	N	Secondary Read-0-to-set
013:012	Configuration CSR, page 143 Configuration CSR	0000	Y	Y
017:014	Downstream I/O Address and Upstream I/O Address Registers, page 144 Downstream I/O Address	Indeterminate	Primary	Y

Table 33. CSR Address Map (Sheet 2 of 5)

Byte Offset (Hex)	Register Name	Reset Value	Write Access	Read Access
01B:018	Downstream I/O Data and Upstream I/O Data Registers, page 145 Downstream I/O Data	Indeterminate	Primary	Primary
01F:01C	Downstream I/O Address and Upstream I/O Address Registers, page 144 Upstream I/O Address	Indeterminate	Secondary	Y
023:020	Downstream I/O Data and Upstream I/O Data Registers, page 145 Upstream I/O Data	Indeterminate	Secondary	Secondary
024	I/O Own Bits Registers, page 145 I/O Own Byte 0	00	N	Primary Read-0-to-set
025	I/O Own Byte 1	00	N	Secondary Read-0-to-set
027:026	I/O CSR, page 146 I/O Own Control and Status	0000	Y	Y
028	Lookup Table Offset Register, page 146 Look-up Table Offset	Indeterminate	Y	Y
02B:029	Reserved	000	N	Y
02F:02C	Lookup Table Data Register, page 147 Look-up Table Data	Indeterminate	Y	Y
033:030	I2O Outbound Post_List Status, page 165 I20 Outbound Post Status	00000000	N	Y
037:034	I2O Outbound Post_List Interrupt Mask, page 165 I20 Outbound Post Mask	00000004	Y	Y
03B:038	I2O Inbound Post_List Status, page 165 I20 Inbound Post Status	00000000	N	Y
03F:03C	I2O Inbound Post_List Interrupt Mask, page 166 I20 Inbound Post Mask	00000004	Y	Y
043:040	I2O Inbound Queue, page 166 I20 Inbound Queue	Indeterminate	Primary	Primary
047:044	I2O Outbound Queue, page 166 I20 Outbound Queue	Indeterminate	Primary	Primary
04B:048	I2O Inbound Free_List Head Pointer, page 167 I20 Inbound Free Head Pointer	Indeterminate	Y	Y
04F:04C	I2O Inbound Post_List Tail Pointer, page 167 I20 Inbound Post Tail Pointer	Indeterminate	Y	Y
053:050	I2O Outbound Free_List Tail Pointer, page 167 I20 Outbound Free Tail Pointer	Indeterminate	Y	Y

Table 33. CSR Address Map (Sheet 3 of 5)

Byte Offset (Hex)	Register Name	Reset Value	Write Access	Read Access
057:054	I2O Outbound Post_List Head Pointer, page 167 I20 Outbound Post Head Pointer	Indeterminate	Y	Y
05B:058	I2O Inbound Post_List Counter, page 168 I20 Inbound Post Counter	00000000	Secondary	Y
05F:05C	I2O Inbound Free_List Counter, page 168 I20 Inbound Free Counter	00000000	Secondary	Y
063:060	I2O Outbound Post_List Counter, page 169 I20 Outbound Post Counter	00000000	Secondary	Y
067:064	I2O Outbound Post_List Counter, page 169 I20 Outbound Free Counter	00000000	Secondary	Y
06B:068	Downstream Memory 0, 2, 3, and Upstream Memory 1 Translated Base Register, page 137 Downstream Memory 0 Translated Base	Indeterminate	Y	Y
06F:06C	Downstream I/O or Memory 1 and Upstream I/O or Memory 0 Translated Base Register, page 136 Downstream I/O or Memory 1 Translated Base	Indeterminate	Y	Y
073:070	Downstream Memory 0, 2, 3, and Upstream Memory 1 Translated Base Register, page 137 Downstream Memory 2 Translated Base	Indeterminate	Y	Y
077:074	Downstream Memory 0, 2, 3, and Upstream Memory 1 Translated Base Register, page 137 Downstream Memory 3 Translated Base	Indeterminate	Y	Y
07B:078	Downstream I/O or Memory 1 and Upstream I/O or Memory 0 Translated Base Register, page 136 Upstream I/O or Memory 0 Translated Base	Indeterminate	Y	Y
07F:07C	Downstream Memory 0, 2, 3, and Upstream Memory 1 Translated Base Register, page 137 Upstream Memory 1 Translated Base	Indeterminate	Y	Y
083:082	Chip Status CSR, page 170 Chip Status CSR	0000	W1TC	Y
085:084	Chip Set IRQ Mask Register, page 170 Chip Set IRQ Mask	FFFF	W1TS	Y
087:086	Chip Clear IRQ Mask Register, page 171 Chip Clear IRQ Mask	FFFF	W1TC	Y
08B:088	Upstream Page Boundary IRQ 0 Register, page 171 Upstream Page Boundary IRQ 0	00000000	W1TC	Y
08F:08C	Upstream Page Boundary IRQ 1 Register, page 172 Upstream Page Boundary IRQ 1	00000000	W1TC	Y

Table 33. CSR Address Map (Sheet 4 of 5)

Byte Offset (Hex)	Register Name	Reset Value	Write Access	Read Access
093:090	Upstream Page Boundary IRQ Mask 0 Register, page 172 Upstream Page Boundary IRQ Mask 0	FFFFFFFF	Y	Y
097:094	Upstream Page Boundary IRQ Mask 1 Register, page 172 Upstream Page Boundary IRQ Mask 1	FFFFFFFF	Y	Y
099:098	Primary Clear IRQ and Secondary Clear IRQ Registers, page 173 Primary Clear IRQ Secondary Clear IRQ	0000	W1TC	Y
09B:09A		0000	W1TC	Y
09D:09C	Primary Set IRQ and Secondary Set IRQ Registers, page 173 Primary Set IRQ Secondary Set IRQ	0000	W1TS	Y
09F:09E		0000	W1TS	Y
0A1:0A0	Primary Clear IRQ Mask and Secondary Clear IRQ Mask Registers, page 174 Primary Clear IRQ Mask Secondary Clear IRQ Mask	FFFF	W1TC	Y
0A3:0A2		FFFF	W1TC	Y
0A5:0A4	Primary Set IRQ Mask and Secondary Set IRQ Mask Registers, page 174 Primary Set IRQ Mask Secondary Set IRQ Mask	FFFF	W1TS	Y
0A7:0A6		FFFF	W1TS	Y
0AB:0A8 0AF:0AC 0B3:0B0 0B7:0B4 0BB:0B8 0BF:0BC 0C3:0C0 0C7:0C4	Scratchpad 0 Through Scratchpad 7 Registers, page 174 Scatchpad 0 Scatchpad 1 Scatchpad 2 Scatchpad 3 Scatchpad 4 Scatchpad 5 Scatchpad 6 Scatchpad 7	Indeterminate	Y	Y
0C9:0C8	ROM Setup Register, page 177 ROM Setup	7E00	Y	Y
0CA	ROM Data Register, page 177ROM Data	Indeterminate	Y	Y
0CB	Reserved	00	N	Y
0CE:0CC	ROM Address Register, page 178 ROM Address	000400	Y	Y
0CF	ROM Control Register, page 178 ROM Control	0000	Y	Y
0D2:0D0	Generic Own Bits Register, page 164Generic Own Bits	TBD	TBD	TBD

Table 33. CSR Address Map (Sheet 5 of 5)

Byte Offset (Hex)	Register Name	Reset Value	Write Access	Read Access
0FF:0D0	Reserved	00000000	N	Y
1FF:100	Upstream Memory 2 Lookup Table, page 147 Upstream Memory 2 Look-up Table	Indeterminate	Y	Y
FFF:200	Reserved	00000000	N	Y

16.4 Address Decoding

16.4.1 Primary and Secondary Address

This section covers pages 16-130 through 16-140 and includes tables Table 34 through Table 60. See Chapter 4 for theory of operation information.

Table 34. Primary CSR and Downstream Memory 0 Bar[a] (Sheet 1 of 2)

- Primary byte offset: 13:10h
- Secondary byte offset: 53:50h

The Primary CSR and Downstream Memory 0 BARs map the 21555 registers into primary memory space. They can specify a downstream memory range for forwarding of memory transactions.

To specify a downstream forwarding range, load the Downstream Memory 0 Setup Register from the optional SROM or the local processor This load must occur before configuration software running on the host processor can access this register.

Local processor access of the setup register should be done before the Primary Lockout Reset Value bit is cleared.

Bit	Name	R/W	Description
0	Space Indicator	R	Indicates the type of address space to setup. When 0, it indicates that memory space is requested.
2:1	Type	R	Indicates size and location of this address space. Reset value is To 00 indicating that this space can be mapped anywhere in 32-bit memory.

Table 34. Primary CSR and Downstream Memory 0 Bar[a] (Sheet 2 of 2)

- Primary byte offset: 13:10h
- Secondary byte offset: 53:50h

The Primary CSR and Downstream Memory 0 BARs map the 21555 registers into primary memory space. They can specify a downstream memory range for forwarding of memory transactions.

To specify a downstream forwarding range, load the Downstream Memory 0 Setup Register from the optional SROM or the local processor This load must occur before configuration software running on the host processor can access this register.

Local processor access of the setup register should be done before the Primary Lockout Reset Value bit is cleared.

Bit	Name	R/W	Description
3	Prefetchable	R	Indicates whether the region is prefetchable. Accesses to the 21555 registers are disconnected after the first data phase. • *When 0,* nonprefetchable memory is requested. • *When 1,* prefetchable memory is requested. • Reset value is 0
11:4	—	R	Returns 0 when read.
31:12	Base Address	R/W	These bits indicate the size of the requested address range and set the base address of the range. • The low 4 KB of this address range map the 21555 CSRs into primary memory space. • The remaining space in this range above 4 KB, if any, specifies a range for downstream forwarding of memory transactions. These bits determine the function of the corresponding bit in this register. • When a bit in the setup register is 0 then the same bit in this register is a read-only bit and always returns 0 when read. • When a bit in the setup register is one (1), the same bit in this register is writable and returns the value last written when read. • When the setup register is written to all zeros, the minimum size of 4 KB is requested (the 21555 CSR access only, no forwarding range). The maximum size of this range is 2 GB; therefore bit [31] is always writable. Reset value is 4 KB of nonprefetchable memory requested.

a. See Chapter 4, “Address Decoding” for more information.

Table 35. Secondary CSR Memory BARs[a] (Sheet 1 of 2)

- Primary byte offset: 53:50h
- Secondary byte offset: 13:10h

Bit	Name	R/W	Description
0	Space Indicator	R	Indicates the type of address space requested. • When a 0, indicate that memory space is requested. • When a one (1),
2:1	Type	R	Indicates size and location of the 21555 memory mapped registers. • When 00, the 21555 registers can be mapped anywhere in 32-bit memory address space.

Table 35. Secondary CSR Memory BARs[a] (Sheet 2 of 2)

• Primary byte offset: 53:50h • Secondary byte offset: 13:10h			
Bit	**Name**	**R/W**	**Description**
3	Prefetchable	R	Indicates if this space is prefetchable. • When a 0, do not use prefetching when reading the 21555 registers.
11:4	—	R	Returns zero.
31:12	Base Address	R/W	Indicate to configuration software the size of the requested memory address range and set the base address of the range. The bits are mappable, and indicates that the 21555 is requesting a 4Kb memory space.

a. The Secondary CSR Memory BARs map the 21555 registers into secondary bus memory space.

Table 36. Primary and Secondary CSR I/O Bars[a]

Offsets	**Primary CSR I/O BAR**	**Secondary CSR I/O BAR**
Primary byte	17:14h	57:54h
Secondary byte	57:54h	17:14h

Bit	**Name**	**R/W**	**Description**
0	Space Indicator	R	Indicates the type of address space that is requested. When a one (1), I/O space is requested.
7:1	Reserved	R	Returns 0.
31:8	Base Address	R/W	Indicates to configuration software the size of the requested I/O address range and set the base address of the range. The bits are mappable, and indicates that the 21555 is requesting a 256 byte I/O space.

a. The Primary and Secondary CSR I/O BARs map the 21555 registers into primary and secondary I/O space, respectively.

Table 37. Downstream I/O or Memory 1 and Upstream I/O or Memory 0 BAR

Offsets	Downstream I/O or Memory 1 BAR	Upstream I/O or Memory 0 BAR
Primary byte	1B:18h	5B:58h
Secondary byte	5B:58h	1B:18h

These registers define forwarding address ranges for downstream or upstream I/O or memory transactions. After reset, they are disabled and return all zeros when read.

This register can request a 64, 128, or 256 bytes I/O space. Hardware does not restrict larger I/O windows or 4 KB to 2 GB. To configure a space use serial preloading or program the Downstream I/O or Memory 1 Setup register or Upstream I/O or Memory 0 Setup register.

Access of the setup registers must be through the local processor before the Primary Lockout Reset Value bit is cleared.

Bit	Name	R/W	Description
0	Space Indicator	R	• When a 0, this BAR is disabled or memory space is requested memory space. • *When a one (1),* I/O space is requested. • Reset value is 0
2:1	Type	R	• When all zeros (0s), I/O space is requested. • When non-zero, memory space is requested and the number equals the size and location of this address range. • Reset value is 00b
3	Prefetchable	R	• When 0, requesting I/O or nonprefetchable memory. • When 1, requesting prefetchable memory space. • Reset value is 0
5:4	—	R	Returns zeros (0s).
31:6	Base Address	R/W	Indicate the size of the requested address range and set the base address of the range. The corresponding setup register determine the function of the corresponding bit in this register. • When a 0, the same bit in this register is a read-only bit and always return 0 when read. • When a one (1), the same bit in this register is writable and return the value last written when read. This BAR is disabled by writing a 0 to bit [31] of the setup register. The minimum size for an I/O address range is 64 bytes and for a memory range is 4 KB. The maximum size is 2 GB. • Reset value is This register is disabled (read only as 0).

Table 38. Downstream Memory 2 and 3 BAR, and Upstream Memory 1 BAR

Offsets	Downstream Memory 2 BAR	Downstream Memory 3 BAR	Upstream Memory 1 BAR
Primary byte	1F:1Ch	23:20h	5F:5Ch
Secondary byte	5F:5Ch	63:60h	1F:1Ch

These registers are similar and are described together.

These registers define address ranges in which memory transactions on the primary interface of the 21555 are forwarded to the secondary interface for the downstream BARs, and in which memory transactions on the secondary interface are forwarded to the primary interface for the upstream BAR.

The corresponding setup registers load from the SROM or the local processor before configuration software running on the host processor can access these registers. Local processor access of the setup registers is before the Primary Lockout Reset Value bit is clear.

Bit	Name	R/W	Description
0	Space Indicator	R	Reads only as 0 to indicate that memory space is requested.
2:1	Type	R	Indicates size and location of this address space. Reset value is 00 to indicate that this space can be mapped anywhere in 32-bit memory.
3	Prefetchable	R	Indicates whether the region is prefetchable. • *When 0,* nonprefetchable memory is requested (or the range is disabled). • *When 1, p*refetchable memory is requested. • Reset value is 0
11:4	—	R	Returns 0.
31:12	Base Address	R/W	These bits are used to indicate the size of the requested address range and to set the base address of the range. Bits [31:12] of the corresponding setup register determine the function of the corresponding bit in this register. • When a bit in the setup register is 0, the same bit in this register is a read only bit and always returns 0 when read. • When a bit in the setup register is one (1), the same bit in this register is writable and returns the value last written when read. This BAR is disabled by writing bit [31] of the setup register to zero. For the Downstream Memory 3 BAR, bit [31] of the Upper 32 Bits setup register must also be 0 to disable that range. The minimum size for this address range is 4 K. The maximum size is 2GB, except for the Downstream Memory 3 BAR which may use 64-bit addressing and have a maximum window size of 2^{63} bytes. • Reset value is Read only as 0 (range is disabled).

Table 39. Upper 32 Bits Downstream Memory 3 Bar

- Primary byte offset: 27:24h
- Secondary byte offset: 67:64h

Bit	Name	R/W	Description
31:0	Base Address	R/W	This register defines the upper 32 bits of a memory range for downstream forwarding of memory transactions. The lower 32 bits are contained in the Downstream Memory 3 BAR. These bits are used to indicate the size of the requested address range and to set the base address of the range. The value of each bit in the Upper 32 Bits Downstream Memory 3 Setup register determines the function of the corresponding bit in this register. • When a bit in the setup register is 0, the same bit in this register is a read-only bit and always return 0 when read. • When a bit in the setup register is one (1), the same bit in this register is writable and returns the value last written when read. This BAR is disabled when bit [31] of the Upper 32 Bits Downstream Memory 3 BAR is 0. The minimum size for this address range is 4 K. The maximum size is 2^{63} bytes. • Reset value is Read only as 0 (range is disabled).

Table 40. Upstream Memory 2 Bar

- Primary byte offset: 63:60h
- Secondary byte offset: 23:20h

This register defines the memory range for upstream forwarding of transactions using lookup table based address translation. All other BARs use direct offset address translation when forwarding transactions.

The size of this register is programmed or disabled by setting the page size in the Chip Control 1 configuration register.

Bit	Name	R/W	Description
0	Space Indicator	R	Reads only as 0 to indicate that memory space is requested.
2:1	Type	R	Indicates size and location of this address space. Reads as 00 to indicate that this space can be mapped anywhere in 32-bit memory.
3	Prefetchable	R	When this address range is enabled, read only as 1h to indicate prefetchable memory. Page entries also may be individually designated as prefetchable or nonprefetchable, where a nonprefetchable entry overrides this prefetchable bit.
13:4	—	R	Read Only. Returns 0 when read.
31:14	Base Address	R/W	These bits are used to indicate the size of the requested address range and to set the base address of the memory range for upstream forwarding using lookup table based address translation. The size of this window is a function of the page size, and can vary from 16 KB to 4 MB increasing by powers of two. The number of writable bits is dependent on the window size, and varies from [31:14] for a 16 KB window to [31:28] for a 256 MB window. Reset value is This address range is disabled (reads only as 0).

Table 41. Downstream I/O or Memory 1 and Upstream I/O or Memory 0 Translated Base Register

Offsets	Downstream I/O or Memory 1 Translated Base	Upstream I/O or Memory 0 Translated Base
Primary byte	9B:98h	A7:A4h
Secondary byte	9B:98h	A7:A4h
CSR byte	06F:06Ch	07B:078h

Bit	Name	R/W	Description
5:0	Reserved	R	Reserved. Returns 0 when read.
31:6	XLAT_BASE	R/W	Contains the translated base address for downstream or upstream transactions whose initiator bus addresses fall into either the Downstream I/O or Memory 1, or Upstream I/O or Memory 0 Base Address range. The number of bits that are used for the translated base is determined by the setup register corresponding to that base address and also matches the number of writable bits in the corresponding BAR. The remaining bits may be written but are ignored when performing address translation. When an I/O or memory transaction is initiated by the 21555 on the target bus, the original base address is replaced with the value contained in this register.

Table 42. Downstream Memory 0, 2, 3, and Upstream Memory 1 Translated Base Register

These registers contain the translated base addresses for their respective downstream and upstream BARs. The base address of the transaction on the initiator bus is replaced by the base address contained in these registers

These registers are also mapped in the 21555 I/O and memory CSR space.

Offsets	Downstream Memory 0 Translated Base	Downstream Memory 2 Translated Base	Downstream Memory 3 Translated Base	Upstream Memory 1 Translated Base
Primary byte	97:94h	9F:9Ch	A3:A0h	AB:A8h
Secondary byte	97:94h	9F:9Ch	A3:A0h	AB:A8h
CSR byte	06B:068h	073:070h	077:074h	07F:07Ch

Bit	Name	R/W	Description
11:0	Reserved	R	Reserved. Returns 0 when read.
31:12	XLAT_BASE	R/W	Contains the translated base address for downstream or upstream transactions whose initiator bus addresses fall into one of the following address ranges: • Downstream Memory 0 (above low 4K boundary) • Downstream Memory 2 • Downstream Memory 3 • Upstream Memory 1 The number of bits that are used for the translated base is determined by the setup register corresponding to that base address and also matches the number of writable bits in the corresponding BAR. The remaining bits can be written but are ignored when performing address translation. When a memory transaction is initiated by the 21555 on the target bus, the original base address is replaced with the value contained in this register.

Table 43. Downstream I/O or Memory 1 and Upstream I/O or Memory 0 Setup Registers

These registers may be preloaded by serial ROM or programmed by the local processor before host configuration.

Offsets	Downstream I/O or Memory 1 Setup	Upstream I/O or Memory 0 Setup
Primary byte	B3:B0h	C7:C4h
Secondary byte	B3:B0h	C7:C4h

Bit	Name	R/W	Description
0	Type Selector	R/(WS)	• *When 0, t*he BAR is requesting memory space, or is disabled. • When 1, the BAR is requesting I/O space. • Reset value is 0
2:1	Type	R/(WS)	Type of space requested. Allowable values: • 00b to indicate that the space requested by the BAR may be located anywhere in memory space, must be used for I/O space • 01b to indicate that memory space must be mapped below a 1MB boundary Other values may have unpredictable results. Reset value is 00h.
3	Prefetchable	R/(WS)	Indicates whether the space requested by the BAR is prefetchable. • *When 0, n*ot prefetchable (required, but not hardware enforced, for I/O space). • When 1, prefetchable. • Reset value is 0
5:4	Reserved	R	Read only as 0.
30:6	Size	R/(WS)	These bits specify the size of the address range requested by the BAR. • When a bit is 1, the corresponding bit in the BAR functions as a readable and writable bit. • When a bit is 0, the corresponding bit in the BAR functions as a read-only bit that always returns zero when read. The *PCI Local Bus Specification, Revision 2.2* states that the maximum value requested for I/O space should not be greater than 256 bytes, although this is not enforced in hardware. When configured as a memory range, bits [11:6] should be set to a 0 as the minimum supported memory range is 4KB. • Reset value is 0 (disabled).
31	BAR_Enable	R/(WS)	BAR enable. • *When 0, t*he corresponding BAR is disabled and reads as 0. • When 1, the corresponding BAR is enabled, with size and type specified by this setup register. • Reset value is 0

Table 44. Downstream Memory 0, 2, 3, and Upstream Memory 1 Setup Registers

These registers are used to program the type and size of their respective upstream and downstream BARs.

Offsets	Downstream Memory 0 Setup	Downstream Memory 2 Setup	Downstream Memory 3 Setup	Upstream Memory 1 Setup
Primary byte	AF:ACh	B7:B4h	BB:B8h	CB:C8h
Secondary byte	AF:ACh	B7:B4h	BB:B8h	CB:C8h

Bit	Name	R/W	Description
0	Type Selector	R	Read only as 0 to indicate memory space is requested by the corresponding memory BAR.
2:1	Type	R/(WS)	Type of space requested. Allowable values are: • 00b to indicate that the space requested by the BAR may be located anywhere in memory space • 01b to indicate that it must be mapped below a 1MB boundary • 10b for Downstream Memory 3 Setup register to request a 64-bit BAR Other values may yield unpredictable results. Reset value is 00b.
3	Prefetchable	R/(WS)	Indicates whether the space requested by the BAR is prefetchable. • When 0, not prefetchable. • When 1, prefetchable. • Reset value is 0
11:4	Reserved	R	Read only as 0.
30:12	Size	R/(WS)	These bits specify the size of the address range requested by the BAR. • When a bit is 1, the corresponding bit in the BAR functions as a readable and writable bit. • When a bit is 0, the corresponding bit in the BAR functions as a read-only bit that always returns zero when read. • Reset value is 0 (disabled), except for Downstream Memory 0 Setup register, whose reset value is 7FFFFh (request 4 KB).
31	BAR_Enable	R/(WS)	BAR enable. Bit [31] of the Downstream Memory 0 Setup register always reads as 1, indicating that the BAR cannot be disabled. When a bus master attempts to write this bit with a 0, the 21555 returns all bits {31:12] of the setup register as 1s (request 4KB). • When the Upper 32 Bits Downstream Memory 3 Setup register bit [31] is a 1, the corresponding BAR is enabled as a 64-bit register, and this bit is part of the size field for the 64-bit BAR. • *When 0, t*he corresponding BAR is disabled and reads as 0, with the exception noted above. • When 1, the corresponding BAR is enabled, with size and type specified by this setup register. • Reset value is 0, except for Downstream Memory 0 Setup register that has a reset value of 1.

Table 45. Upper 32 Bits Downstream Memory 3 Setup Register

This register may be preloaded by serial ROM or programmed by the local processor before host configuration.

- Primary byte offset: BF:BCh
- Secondary byte offset: BF:BCh

Bit	Name	R/W	Description
30:0	Size	R/(WS)	These bits specify upper 32 bits of the size of the address range requested by Downstream Memory 3 BAR. • When a bit is 1, the corresponding bit in Downstream Memory 3 BAR functions as a readable and writable bit. • When a bit is 0, the corresponding bit in Downstream Memory 3 BAR functions as a read-only bit that always returns 0 when read. These bits must be set to a non-zero value only when bits [2:1] of Downstream Memory 3 BAR are set to 10b (this is not enforced in hardware). • Reset value is 0
31	BAR_Enable	R/(WS)	64-bit Downstream Memory 3 BAR enable. • *When 0, t*he Downstream Memory 3 64-bit BAR is disabled (but may still be a 32-bit BAR). • When 1, the Downstream Memory 3 BAR is enabled as a 64-bit BAR. • Reset value is 0 (disabled)

16.4.2 Configuration Transaction Generation Registers

All of these registers are mapped into the 21555 configuration space and described in Section 16.4.2. Note that the 21555 initiates a transaction only when the Configuration Data registers are accessed at these locations using I/O reads and writes.

The Downstream Configuration Data Register and the Upstream Configuration Data Register are treated as reserved registers for all memory accesses.

Table 46. Downstream and Upstream Configuration Address Registers

This section describes both the downstream and upstream versions of the registers. These registers are also mapped in memory and I/O space.

Offsets	Downstream Configuration Address	Upstream Configuration Address
Primary byte	83:80h	8B:88h (Reserved)
Secondary byte	83:80h (Reserved)	8B:88h
CSR Space	003:000h	00B:008h

Bit	Name	R/W	Description
31:0	CFG_ADDR (CA)	DCA: R/(WP) UCA: R/(WS)	This register contains the address for a configuration transaction to be generated on the target bus. The address is driven exactly as written in this register. This register should be written before the corresponding Downstream or Upstream Configuration Data register is accessed. Once the Downstream or Upstream Configuration Data register is accessed, the transaction is initiated on the secondary or primary bus, respectively. When the semaphore method is used, a master should not write to this register unless the master has successfully read a 0 from the Downstream or Upstream Configuration Own bit. The Downstream Configuration Address register cannot be written from the secondary interface. The Upstream Configuration Address register cannot be written from the primary interface.

.

Table 47. Downstream Configuration Data and Upstream Configuration Data Registers

These registers are also mapped in memory and I/O space. This register is treated as a reserved register for all memory accesses.

Offsets	Downstream Configuration Data	Upstream Configuration Data
Primary byte	87:84h	8F:8Ch (Reserved)
Secondary byte	87:84h (Reserved)	8F:8Ch
CSR Space	007:004h	00F:00Ch

Bit	Name	R/W	Description
31:0	CFG_DATA (CD)	DCD: R/(WP) UCD: R/(WS)	This register contains the write data driven or the read data returned from a configuration transaction initiated by the 21555. The Downstream or Upstream Configuration Address register contains the address for this transaction, depending on the direction of the transaction. The transaction is initiated when this register is written (for a configuration write) or read (for a configuration read) and the corresponding Configuration Control bit is a one. The byte enables used for this register access are the same byte enables used for the transaction driven on the target bus. A target retry is returned to the initiator until the transaction has been completed on the target bus. When the semaphore method is used, a master should not write to this register unless the master has successfully read a 0 from the Downstream or Upstream Configuration Own bit. The Downstream Configuration Data register is reserved when accessed from the secondary interface, or on either interface when the Downstream Configuration Enable bit is not set. The Upstream Configuration Data register is reserved when accessed from the primary interface, or on either interface when the Upstream Configuration Enable bit is not set.

Table 48. Configuration Own Bits Register

This register is also mapped in memory and I/O space.

- Primary byte offset: 91:90h
- Secondary byte offset: 91:90h
- CSR byte offset 011:010h.

Bit	Name	R/W	Description
0	Downstream Configuration Own Bit	R0TS (P) R(S)	Indicates ownership of the Downstream Configuration Address and Downstream Configuration Data registers. • *When 0, d*ownstream Configuration Address and Downstream Configuration Data registers are not owned. When read as a 0 from the primary interface, this bit is subsequently set to a 1 by the 21555 when the Downstream Configuration Control bit is a 1. • When 1, a master owns Downstream Configuration Address and Downstream Configuration Data registers. When this semaphore method is used, other masters should not attempt to access these registers when this bit is a 1. This bit is automatically cleared once the configuration transaction has completed on the initiator bus. • Reset value is 0.

Table 48. Configuration Own Bits Register

7:1	Reserved	R	Read only as 0.
8	Upstream Configuration Own Bit	R0TS (S) R(P)	Indicates ownership of the Upstream Configuration Address and Upstream Configuration Data registers. • *When 0, u*pstream Configuration Address and Upstream Configuration Data registers are not owned. When read as a 0 from the secondary interface, this bit is subsequently set to a 1 by the 21555 if the Upstream Configuration Control bit is a 1. • When 1, a master owns Upstream Configuration Address and Upstream Configuration Data registers. When this semaphore method is used, other masters should not attempt to access these registers when this bit is a 1. This bit is automatically cleared once the configuration transaction has completed on the initiator bus. • Reset value is 0.
15:9	Reserved	R	Read only as 0.

Table 49. Configuration CSR (Sheet 1 of 2)

This register is also mapped in memory and I/O space.
- Primary byte offset: 93:92h
- Secondary byte offset: 93:92h
- CSR byte offset: 013:012h

Bit	Name	R/W	Description
0	Downstream Configuration Own Status	R	Provides the current value of the Downstream Configuration Own bit. This bit has no side effects when read.
1	Downstream Configuration Control	R/W	Enables the 21555 to perform downstream indirect configuration transactions. • *When 0, t*he 21555 will not initiate a configuration transaction on the secondary interface when the Downstream Configuration Data register is accessed. The Downstream Configuration Data register is treated as a reserved register. • When 1, the 21555 is enabled to perform downstream configuration transactions when the Downstream Configuration Data register is accessed. • Reset value is 0
2	Downstream Self-Response Enable	R/W	Controls the 21555 ability to respond to a configuration transaction that it generates as a master. • *When 0, t*he 21555 does not respond to configuration transactions that it generates. • When 1, the 21555 does not respond to configuration transactions that it generates as a master. • Reset value is 0
7:3	Reserved	R	Reserved. Returns 0 when read.
8	Upstream Configuration Own Status	R	Provides the current value of the Upstream Configuration Own bit. This bit has no side effects when read.

Table 49. Configuration CSR (Sheet 2 of 2)

This register is also mapped in memory and I/O space.
- Primary byte offset: 93:92h
- Secondary byte offset: 93:92h
- CSR byte offset: 013:012h

Bit	Name	R/W	Description
9	Upstream Configuration Control	R/W	Enables the 21555 to perform upstream indirect configuration transactions. • *When 0, t*he 21555 will not initiate a configuration transaction on the primary interface when the Upstream Configuration Data register is accessed. The Upstream Configuration Data register is treated as a reserved register. • When 1, the 21555 is enabled to perform upstream configuration transactions when the Upstream Configuration Data register is accessed. • Reset value is 0
10	Upstream Self-Response Enable	R/W	Controls the 21555 ability to respond to a configuration transaction that it generates as a master. • *When 0, t*he 21555 does not respond to configuration transactions that it generates. These transaction end in master abort. • When 1, the 21555 does not respond to configuration transactions that it generates as a master. • Reset value is 0
15:11	Reserved	R	Reserved. Reads only as 0.

Table 50. Downstream I/O Address and Upstream I/O Address Registers

The Downstream I/O Address register is used for I/O transactions to be initiated on the secondary bus, and the Upstream I/O Address register is used for I/O transactions to be initiated on the primary bus. The downstream register can be written from the primary interface only and the upstream register can be written from the secondary interface only.

Offset	Downstream I/O Address	Upstream I/O Address
Byte	017:014h	1F:1Ch

Bit	Name	R/W	Description
31:0	IO_ADDR (IA)	DIA: R/(WP) UIA: R/(WS)	This register contains the address for an I/O transaction to be generated on the target bus. The address is driven exactly as written in this register. This register should be written before the Downstream or Upstream I/O Data register is accessed. Once the Downstream or Upstream I/O Data register is written or read, the transaction is initiated on the secondary bus. When the semaphore method is used, a master should not write to this register unless the master has successfully read a 0 from the Downstream or Upstream I/O Own bit. Reset value is 0

Table 51. Downstream I/O Data and Upstream I/O Data Registers

The Downstream I/O Data register is used for I/O transactions to be initiated on the secondary bus, and the Upstream I/O Data register is used for I/O transactions to be initiated on the primary bus. The downstream register can be written from the primary interface only and the upstream register can be written from the secondary interface only. A downstream transaction is initiated by a primary interface I/O register access only and an upstream transaction is initiated by a secondary interface I/O access only.

Offsets	Downstream I/O Data	Upstream I/O Data
Byte	01B:018h	023:020h

Bit	Name	R/W	Description
31:0	IO_DATA (ID)	DID: R/(WP) UID: R/(WS) Res (Mem)	This register contains the write data driven or the read data returned from an I/O transaction initiated on the target bus. The Downstream or Upstream I/O Address register contains the address for this transaction. The transaction is initiated when this register is written (for an I/O write) or read (for an I/O read) using an I/O transaction. This register is treated as a reserved register for all memory accesses. The byte enables used for this register access are the same byte enables used for the transaction driven on the target bus. A target retry is returned to the initiator until the transaction has been completed on the target bus. Reset value is 0

Table 52. I/O Own Bits Registers

- Byte Offset: 025:024h

Bit	Name	R/W	Description
0	Downstream I/O Own Bit	R0TS (P) R (S)	Indicates ownership of the Downstream I/O Address and Downstream I/O Data registers. • *When 0,* downstream I/O Address and Downstream I/O Data registers are not owned. When read as a 0 from the primary interface, this bit is subsequently set to a 1 by the 21555. • When 1, downstream I/O Address and Downstream I/O Data registers are owned by a master. When the semaphore method is used, other masters should not attempt to access these registers when this bit is a 1. This bit is automatically cleared once the I/O transaction has completed on the initiator bus. • Reset value is 0.
7:1	Reserved	R	Read only as 0.
8	Upstream I/O Own Bit	R0TS (S) R (P)	Indicates ownership of the Upstream I/O Address and Upstream I/O Data registers. • *When 0,* upstream I/O Address and Upstream I/O Data registers are not owned. When read as a 0 from the secondary interface, this bit is subsequently set to a 1 by the 21555. • When 1, upstream I/O Address and Upstream I/O Data registers are owned by a master. When the semaphore method is used, other masters should not attempt to access these registers when this bit is a 1. This bit is automatically cleared once the I/O transaction has completed on the initiator bus. • Reset value is 0.
15:9	Reserved	R	Read only as 0.

Table 53. I/O CSR

• Byte Offset: 027:026h			
Bit	**Name**	**R/W**	**Description**
0	Downstream I/O Own Bit Status	R	This bit reflects the status of the Secondary Own bit used for generating I/O transaction on the secondary bus. • *When 0, t*he Downstream I/O Address and Downstream I/O Data registers are not owned. • When 1, the Downstream I/O Address and Downstream I/O Data registers are owned by a master.
1	Downstream I/O Control	R/W	Enables the 21555 to perform downstream indirect I/O transactions. • *When 0, t*he 21555 will not initiate a I/O transaction on the secondary interface when the Downstream I/O Data register is accessed. The Downstream I/O Data register is treated as a reserved register. • When 1, the 21555 is enabled to perform downstream I/O transactions when the Downstream I/O Data register is accessed with an I/O transaction. • Reset value is 0
7:2	Reserved	R	Reserved. Read only as 0.
8	Upstream I/O Own Bit Status	R	This bit reflects the status of the Primary Own bit used for generating I/O transaction on the Primary bus. • *When 0, t*he Upstream I/O Address and Upstream I/O Data registers are not owned. • When 1, the Upstream I/O Address and Upstream I/O Data registers are owned by a master.
9	Upstream I/O Control	R/W	Enables the 21555 to perform upstream indirect I/O transactions. • *When 0, t*he 21555 will not initiate an I/O transaction on the primary interface when the Upstream I/O Data register is accessed. The Upstream I/O Data register is treated as a reserved register. • When 1, the 21555 is enabled to perform upstream I/O transactions when the Upstream I/O Data register is accessed with an I/O transaction. • Reset value is 0
15:10	Reserved	R	Reserved. Read only as 0.

Table 54. Lookup Table Offset Register

Table 54 and Table 55 are registers that provide a method for the lookup table to be accessed using I/O transactions, although memory transactions can use either this mechanism or direct access of the lookup table.

• Byte Offset: 028h

Bit	**Name**	**R/W**	**Description**
7:0	LUT_OFFSET	R/W	This register contains the byte offset of the Lookup Table entry to be accessed. The access is initiated when the Lookup Table Data register is either read or written. This register should be written before the Lookup Table Data register is accessed.

Table 55. Lookup Table Data Register

Table 54 and Table 55 are registers that provide a method for the lookup table to be accessed using I/O transactions, although memory transactions can use either this mechanism or direct access of the lookup table.

The lookup table is not byte-writable; byte enables are ignored.

- Byte Offset: 02F:02Ch

Bit	Name	R/W	Description
31:0	LUT_DATA	R/W	This register contains the data written to or read from the Lookup Table at the offset given in the Lookup Table Offset register. When this register is written, the value written is written to the specified Lookup Table entry. When this register is read, the value returned reflects the data read from the specified Lookup Table entry. The following fields are defined: • bit 0: valid bit • bits 2:1: reserved (read only as 0) • bit 3: prefetchable • bits 7:4: reserved (read only as 0) • bits 17:8: translated base or reserved (based on page size) • bits 31:18: translated base The lookup table is not reset and therefore powers up to an indeterminate value.

Table 56. Upstream Memory 2 Lookup Table

The lookup table is not byte-writable; byte enables are ignored.

- Byte Offsets: 1FF:100h

Bit	Name	R/W	Description
	M2LUT	R/W	Contains the lookup table for the Upstream Memory 2 Base Address range. Each entry in the lookup table is 4 bytes wide. The top 16 to 24 bits, depending on the page size, of each entry are used to replace the page address of upstream memory transactions falling inside the Upstream Memory 2 BAR. The bottom four bits are control bits as described in Section 4.3.3.

16.5 PCI Registers

This section covers pages 16-147 through 16-165 and Table 57 through Table 80. See Chapter 3 or Chapter 5 for theory of operation information.

16.5.1 Configuration Registers

The registers described in this section are shared between the primary and secondary interfaces.

Table 57. Primary Interface Configuration Space Address Map

Byte 3	Byte 2	Byte 1	Byte 0	Primary Offset	Secondary Offset
Device ID[1]		Vendor ID[1]		00h	40h
Primary Status		Primary Command		04h	44h
Primary Class Code[2]			Revision ID[1]	08h	48h
BIST[1,2]	Header Type[1]	Primary MLT	Primary CLS	0Ch	4Ch
Subsystem ID[1,2]		Subsystem Vendor ID[1,2]		2Ch	6Ch
Reserved			Cap_Ptr[1]	34h	74h
Primary Max_Lat[2]	Primary Min_Gnt[2]	Primary Interrupt Pin	Primary Interrupt Line	3Ch	7Ch

1. Primary and secondary configuration registers are shared.
2. Register or a portion of the register may be preloaded using the serial ROM interface.

Table 58. Secondary Interface Configuration Space Address Map

Byte 3	Byte 2	Byte 1	Byte 0	Primary Offset	Secondary Offset
Device ID[1]		Vendor ID[1]		40h	00h
Secondary Status		Secondary Command		44h	04h
Secondary Class Code[1]			Revision ID[1]	48h	08h
BIST[1,2]	Header Type[1]	Secondary MLT	Secondary CLS	4Ch	0Ch
Subsystem ID[1,2]		Subsystem Vendor ID[1,2]		6Ch	2Ch
Reserved			Cap_Ptr[1]	74h	34h
Secondary Max_Lat[2]	Secondary Min_Gnt[2]	Secondary Interrupt Pin	Secondary Interrupt Line	7Ch	3Ch

1. Primary and secondary configuration registers are shared.
2. Register or a portion of the register may be preloaded using the serial ROM interface.

Table 59. Vendor ID Register

- Primary byte offset: 01:00h and 41:40h
- Secondary byte offset: 41:40h and 01:00h

Bit	Name	R/W	Description
15:0	Vendor ID	R	The Vendor ID identifies Intel® as the vendor of this device and is internally hardwired to be 8086 hex.

Table 60. Device ID Register

- Primary byte offset: 03:02h and 43:42h
- Secondary byte offset: 43:42h and 03:02h

Bit	Name	R/W	Description
15:0	Device ID	R	Device ID identifies this device as the 21555 and is internally hardwired to be B555h.

16.5.2 Primary and Secondary Command Registers

The register types in this section have separate registers for the primary and secondary interfaces. However, the register description is given once, and applies to both the primary and secondary configuration registers. The primary register controls behavior on the primary interface only, and the secondary register controls behavior on the secondary interface only.

Table 61. Primary and Secondary Command Registers (Sheet 1 of 2)

Offsets	Primary Command	Secondary Command
Primary byte	05:04h	45:44h
Secondary byte	45:44h	05:04h

Bit	Name	R/W	Description
0	I/O Space Enable	R/W	Controls response to I/O transactions on the corresponding interface. • When 0, the 21555 does not respond to I/O transactions. • When 1, the 21555 response to I/O transactions is enabled. • Reset value is 0
1	Memory Space Enable	R/W	Controls response to memory transactions on the corresponding interface. • When 0, the 21555 does not respond to memory transactions. • When 1, the 21555 response to memory transactions is enabled. • Reset value is 0.
2	Master Enable	R/W	Controls 21555's ability to initiate memory and I/O transactions on the corresponding interface. Initiation of configuration transactions is not affected. • When 0, the 21555 does not initiate memory or I/O transactions. • When 1, the 21555 is enabled to operate as an initiator. • Reset value is 0.
3	Special Cycle Enable	R	The 21555 ignores special cycle transactions, so this bit is read only and returns 0.
4	Memory Write and Invalidate Enable	R/W	This bit controls the ability of the 21555 to generate Memory Write and Invalidate (MWI) bus commands as a master on the corresponding interface. • When 0, Disables use of MWI bus commands (uses Memory Write commands instead). • When 1, Enables use of MWI bus commands. • Reset value is 0
5	VGA Snoop Enable	R	Reads only as 0 to indicate the 21555 does not respond to VGA palette writes.
6	Parity Error Response	R/W	Controls the response of the 21555 when a parity error is detected on the corresponding interface. • When 0, the 21555 does not assert **PERR#**, nor does it set the Data Parity Reported bit in the appropriate Primary or Secondary Status registers. The 21555 does not report address parity errors by asserting **SERR#**. • When 1, the 21555 drives **PERR#** and conditionally sets the Data Parity Reported bit in the Primary or Secondary Status register when a data parity error is detected. The 21555 allows **SERR#** assertion when address parity errors are detected. • Reset value is 0.

Table 61. Primary and Secondary Command Registers (Sheet 2 of 2)

Offsets	Primary Command	Secondary Command
Primary byte	05:04h	45:44h
Secondary byte	45:44h	05:04h

Bit	Name	R/W	Description
7	Wait Cycle Control	R	Reads as zero to indicate the 21555 does not perform address or data stepping.
8	SERR# Enable	R/W	Controls the enable for **SERR#** on the corresponding interface. • When 0, **SERR#** cannot be driven by the 21555. • When 1, **SERR#** may be driven low by the 21555 under the conditions described in Chapter 12. • Reset value is 0.
9	Fast Back-to-Back Enable	R/W	Controls the ability of the 21555 to generate fast back-to-back transactions on the corresponding bus. • When 0, the 21555 does not generate back-to-back transactions. • When 1, the 21555 is enabled to generate back-to-back transactions. • Reset value is 0.
15:10	Reserved	R	Reserved. Returns 0 when read.

Table 62. Primary and Secondary Status Registers (Sheet 1 of 2)

The bits described in Table 62 reflect the status of the 21555 primary interface for the Primary Status register, and of the secondary interface for the Secondary Status register. W1TC indicates that writing a 1 to that bit clears the bit to 0. Writing a 0 has no effect.

Offsets	Primary Status	Secondary Status
Primary byte	07:06h	47:46h
Secondary byte	47:46h	07:06h

Bit	Name	R/W	Description
3:0	Reserved	R	Reserved. Returns 0 when read.
4	ECP Support	R	Enhanced Capabilities Support Indicator. Reads as 1 to indicate that the Enhanced Capabilities Port is supported.
5	66 MHz Capable	R	66 MHz Capable Indication. Reads as 0 to indicate that the corresponding interface operates at a maximum frequency of 33 MHz. Reads as 1 to indicate that the corresponding interface is 66 MHz capable. Product derivatives hardcode this to either 0 or 1.
6	Reserved	R	Reserved. Returns 0 when read.
7	Fast Back-to-Back Capable	R	Reads as 1 to indicate that the 21555 is able to respond to fast back-to-back transactions on the corresponding interface.

Table 62. Primary and Secondary Status Registers (Sheet 2 of 2)

The bits described in Table 62 reflect the status of the 21555 primary interface for the Primary Status register, and of the secondary interface for the Secondary Status register. W1TC indicates that writing a 1 to that bit clears the bit to 0. Writing a 0 has no effect.

Offsets	Primary Status	Secondary Status
Primary byte	07:06h	47:46h
Secondary byte	47:46h	07:06h

Bit	Name	R/W	Description
8	Data Parity Detected	R/ W1TC	This bit is set to a 1 when all of the following are true: • The 21555 is a master on the corresponding bus. • **PERR#** is detected asserted for writes or a parity error is detected for reads. • Parity Error Response bit is set in the Primary or Secondary Command register. Reset value is 0.
10:9	DEVSEL# timing	R	Indicates slowest response to a non configuration command on the corresponding interface. Reads as 01b to indicate that the 21555 responds no slower than with medium timing.
11	Signaled Target Abort	R/ W1TC	This bit is set to a 1 when the 21555 is acting as a target on the corresponding bus and returns a target abort to the initiator. Reset value is 0.
12	Received Target Abort	R/ W1TC	This bit is set to a 1 when the 21555 is acting as an initiator on the corresponding bus and receives a target abort. Reset value is 0.
13	Received Master Abort	R/ W1TC	This bit is set to a 1 when the 21555 is acting as an initiator on the corresponding bus and detects a master abort. Reset value is 0.
14	Signaled System Error	R/ W1TC	This bit is set to a 1 when the 21555 has asserted **SERR#** on the corresponding bus. Reset value is 0.
15	Detected Parity Error	R/ W1TC	This bit is set to a 1 when the 21555 detects an address or data parity error on the corresponding interface. Reset value is 0.

Table 63. Revision ID (Rev ID) Register

- Primary byte offset: 08h and 48h
- Secondary byte offset: 08h and 48h

Bit	Name	R/W	Description
7:0	Revision ID	R	This register indicates the revision number of this device. The initial revision reads as 0. Subsequent revisions increment by 1.

Table 64. Primary and Secondary Class Code Registers

These registers may be preloaded through the serial ROM. The Primary Class Code register may also be programmed by the local processor before host configuration.

Offsets	Primary Class Code	Secondary Class Code
Primary byte	0B:09h	4B:49h
Secondary byte	4B:49h	0B:09h

Bit	Name	R/W	Description
7:0	Prog IF (PIF)	PPIF: R/(WS) SPIF: R	Reads as zero.
15:8	Sub-Class Code (SCC)	PSCC: R/(WS) SSCC: R	Reads as 80 hex to indicate that this bridge device is classified as “other”.
23:16	Base Class Code (BCC)	PBCC:R/(WS) SBCC: R	Reads as 06 hex to indicate device is a bridge device.

Table 65. Primary and Secondary Cache Line Size Registers

Offsets	Primary Cache Line Size	Secondary Cache Line Size
Primary byte	0Ch	4Ch
Secondary byte	4Ch	0Ch

Bit	Name	R/W	Description
7:0	Cache Line Size	R/W	Designates the cache line size for the corresponding interface in units of 32-bit Dwords. Used for prefetching memory reads and for terminating MWIs. Valid cache line sizes are 8, 16, and 32 Dwords. When the cache line size is set to any other value, the 21555 uses the same behavior as when the cache line size is set to 8. Reset value is 00h.

Table 66. Primary Latency and Secondary Master Latency Timer Registers

Offsets	Primary MLT	Secondary MLT
Primary byte	0Dh	4Dh
Secondary byte	4Dh	0Dh

Bit	Name	R/W	Description
7:0	Master Latency Timer	R/W	Master latency timer for the corresponding interface. Indicates the number of PCI clock cycles from the assertion of **FRAME#** to the expiration of the timer when the 21555 is acting as a master. All bits are writable, resulting in a granularity of 1 PCI clock cycle. • When 0, the 21555 relinquishes the bus after the first data transfer when the 21555's PCI bus grant has been deasserted. • Reset value is 00h.

Table 67. Header Type Register

- Primary byte offset: 0Eh and 4Eh
- Secondary byte offset: 0Eh and 4Eh

Bit	Name	R/W	Description
7:0	Header Type	R	Defines the layout of addresses 00h through 3Fh in configuration space. Reads as 00h indicating a Type 0 header format.

Table 68. BiST Register

The 21555 does not implement self-test internally and does not directly use the BiST register. However, some form of self-test may be desired in the subsystem so mechanisms are provided by the 21555 to support vendor-specific usage of the BiST register. The default value of this register is 00h after reset assertion, which indicates that BiST is not supported. However, after reset the 21555 allows this field to be automatically preloaded with a value from the serial ROM (when attached) or programmed via the secondary interface by the local process

- Primary byte offset: 0Fh and 4Fh
- Secondary byte offset: 0Fh and 4Fh

Bit	Name	R/W	Description
3:0	Completion Code	R/(WS)	The completion code can only be written by the secondary interface (at secondary offset 0Fh or offset 4Fh). A Completion Code value of 0h indicates that the device passed its self-test. Any non-zero value in the Completion Code indicates that the device failed its self-test.
5:4	Reserved	R	Reserved. Read only as 0.
6	Self Test	R/W	This bit can be written via the primary interface or secondary interface configuration registers. Configuration code running on the host processor sets this bit to 1 to invoke self-test. The local processor (or some other device) on the secondary interface clears this bit to 0 to indicate the completion of the self-test (after first updating the Completion Code bit field).
7	BiST Supported	R/(WS)	This bit can be written by the secondary interface (at secondary offset 0Fh or offset 4Fh) or it may be preloaded using the serial ROM. A value of 1 indicates to configuration software that the device supports self-test. A value of 0 indicates that the device does not support self-test.

Table 69. Subsystem Vendor ID Register

• Primary byte offset: 2D:2Ch and 6D:6Ch • Secondary byte offset: 6D:6Ch and 2D:2Ch			
Bit	**Name**	**R/W**	**Description**
15:0	Subsystem Vendor ID	R/(WS)	Identifies the vendor of the add-in card or subsystem. This register is initialized by either the local processor or by serial ROM preload.

Table 70. Subsystem ID Register

• Primary byte offset: 2F:2Eh and 6F:6Eh • Secondary byte offset: 6F:6Eh and 2F:2Eh			
Bit	**Name**	**R/W**	**Description**
15:0	Subsystem ID	R/(WS)	Identifies the vendor-specific device ID for subsystem. This register is initialized by either the local processor or by serial ROM preload.

Table 71. Enhanced Capabilities Pointer Register

Offsets	**ECP**
Primary byte	34h and 74h
Secondary byte	34h and 74h

Bit	**Name**	**R/W**	**Description**
7:0	ECP	R	Pointer to the first set of ECP registers. Returns DCh to indicate that the first set of ECP registers begins at configuration offset DCh. For the 21555, this points to the Power Management registers. Reset value is DCh

Table 72. Primary and Secondary Interrupt Line Registers

Offsets	**Primary Interrupt Line**	**Secondary Interrupt Line**
Primary byte	3Ch	7Ch
Secondary byte	7Ch	3Ch

Bit	**Name**	**R/W**	**Description**
7:0	Interrupt Line	R/W	This register is used to communicate interrupt line routing information for the corresponding interface. This register must be initialized by initialization code so a default state after reset assertion is not specified. Initialization code writes this register with a value indicating to which input of the system interrupt controller the 21555 bus interrupt signal pin **INTA#** is connected.

Table 73. Primary and Secondary Interrupt Pin Registers

Offsets	Primary Interrupt Pin	Secondary Interrupt Pin
Primary byte	3Dh	7Dh
Secondary byte	7Dh	3Dh

Bit	Name	R/W	Description
7:0	Interrupt Pin	R	This register indicates which PCI interrupt pin the 21555 uses on the corresponding bus. This is a read-only register and always returns 1 when read indicating that the 21555 uses **INTA#**.

Table 74. Primary and Secondary Minimum Grant Registers

These registers may be preloaded through the serial ROM. The Primary Minimum Grant register may also be programmed by the local processor.

Offsets	Primary Minimum Grant	Secondary Minimum Grant
Primary byte	3Eh	7Eh
Secondary byte	7Eh	3Eh

Bit	Name	R/W	Description
7:0	MIN_GNT (MG)	PMG: R/(WS) SMG: R	Specifies how long of a burst period the 21555 needs on the corresponding bus in units of 1/4 µsec. Reads as 0 before preload.

Table 75. Primary and Secondary Maximum Latency Registers

These registers may be preloaded through the serial ROM. The Primary Maximum Latency register may also be programmed by the local processor.

Offsets	Primary Maximum Latency	Secondary Maximum Latency
Primary byte	3Fh	7Fh
Secondary byte	7Fh	3Fh

Bit	Name	R/W	Description
7:0	MAX_LAT (ML)	PML: R/(WS) SML: R	Specifies how often the 21555 needs to gain access to the corresponding bus in units of 1/4 µsec. Reads as 0 before preload.

16.5.3 Device-Specific Control and Status Registers

This section contains information about the device-specific control and status registers.

Table 76. Device-Specific Control and Status Address Map

Byte 3	Byte 2	Byte 1	Byte 0	Primary Offset	Secondary Offset
Chip Control 1		Chip Control 0		CCh	CCh
		Chip Status		D0h	D0h

Table 77. Chip Control 0 Register (Sheet 1 of 4)

This register may be preloaded by serial ROM or programmed by the local processor before host configuration.

- Primary byte offset: CD:CCh
- Secondary byte offset: CD:CCh

Bit	Name	R/W	Description
0	Master Abort Mode	R/W	Controls the 21555's behavior on the initiator bus when a master abort termination occurs in response to a delayed transaction initiated by the 21555 on the target bus. • When 0, the 21555 asserts **TRDY#** in response to a delayed transaction, and returns FFFFFFFFh if a read. For posted writes, **SERR** is not asserted on the initiator bus. • When 1, the 21555 returns a target abort in response to a delayed transaction. For posted writes, **SERR** will be asserted (if otherwise enabled) on the initiator bus. • Reset value is 0
1	Memory Write Disconnect Control	R/W	Controls the disconnect boundary for memory writes. This bit does not apply to MWI commands. • When 0, the 21555 disconnects memory writes either on an aligned 4 KB boundary, a page boundary less than 4 KB (Upstream Memory Range 2 only) or when the posted write queue is full. • When 1, the 21555 disconnects memory write on an aligned cache line boundary, or when the posted write queue is full. • Reset value is 0
2	Primary Master Timeout	R/W	Sets the maximum number of PCI clock cycles that the 21555 waits for an initiator on the primary bus to repeat a delayed transaction request. The counter starts when the delayed transaction completion is ready to be returned to the initiator. When the initiator has not repeated the transaction at least once before the counter expires, the 21555 discards the delayed transaction from its queues. • When 0, the primary master timeout counter is 2^{15} PCI clock cycles, or 0.983 ms for a 33-MHz bus. • When 1, the value is 2^{10} PCI clock cycles, or 30.7 µs for a 33-MHz bus. • Reset value is 0

Table 77. Chip Control 0 Register (Sheet 2 of 4)

This register may be preloaded by serial ROM or programmed by the local processor before host configuration.

- Primary byte offset: CD:CCh
- Secondary byte offset: CD:CCh

Bit	Name	R/W	Description
3	Secondary Master Timeout	R/W	Sets the maximum number of PCI clock cycles that the 21555 waits for an initiator on the secondary bus to repeat a delayed transaction request. The counter starts when the delayed transaction completion is ready to be returned to the initiator. When the initiator has not repeated the transaction at least once before the counter expires, the 21555 discards the delayed transaction from its queues. • When 0, the secondary master timeout counter is 2^{15} PCI clock cycles, or.983ms for a 33-MHz bus. • When 1, the value is 2^{10} PCI clock cycles, or 30.7 µs for a 33-MHz bus. • Reset value is 0
4	Primary Master Timeout Disable	R/W	Disables the primary master timeout counter. • When 0, the primary master timeout counter is enabled and uses the value specified by the Primary Master timeout bit. • When 1, the primary master timeout counter is disabled. The 21555 waits indefinitely for a primary bus initiator to repeat a delayed transaction. • Reset value is 0
5	Secondary Master Timeout Disable	R/W	Disables the secondary master timeout counter. • When 0, the secondary master timeout counter is enabled and uses the value specified by the Secondary Master Timeout bit. • When 1, the secondary master timeout counter is disabled. The 21555 waits indefinitely for a secondary bus initiator to repeat a delayed transaction. • Reset value is 0
6	Delayed Transaction Order Control	R/W	Controls how the 21555 initiates delayed transactions on the target bus. • When 0, the 21555 uses a round-robin arbitration scheme to determine which transaction is attempted. After receiving a target retry in response to a delayed transaction, the 21555 can initiate a different queued delayed transaction. • When 1, When a target retry is received in response to a delayed transaction, the 21555 continues to attempt that same transaction until a response other than target retry is received. The 21555 does not initiate other delayed transactions until the above condition is satisfied. • Reset value is *0*.
7	SERR# Forward Enable	R/W	**SERR#** forward enable. When 0, the 21555 does not assert **p_serr_l** as a result of **s_serr_l** assertion. When 1, the 21555 asserts **p_serr_l** when **s_serr_l** is detected asserted and the primary **SERR#** Enable bit is set. Reset value is 0

Table 77. Chip Control 0 Register (Sheet 3 of 4)

This register may be preloaded by serial ROM or programmed by the local processor before host configuration.

- Primary byte offset: CD:CCh
- Secondary byte offset: CD:CCh

Bit	Name	R/W	Description
8	Upstream DAC Prefetch Disable	R/W	Controls prefetching for upstream dual address transactions using the memory read bus command. When 0, prefetching is performed for upstream DAC memory reads. When 1, upstream DACs using the memory read bus command are not prefetched; transactions are limited to a single Dword and byte enables are preserved. Reset value is 0
9	Multiple Device Enable	R/W	Enables multiple devices to be attached to the ROM interface. • When 0, only the parallel and serial ROM can be attached to the ROM interface. The PROM (PROM) chip select is driven on the **pr_cs_l** pin. • When 1, multiple devices may be attached to the ROM interface. All chip selects with the exception of the serial ROM are decoded from the upper address lines of the ROM interface. • Reset value is 0
10	Primary Access Lockout	R/(WS)	This bit prevents the primary bus from accessing configuration space. This allows the local processor to access the 21555 registers before the host processor accesses them. This bit can be written from the secondary interface only. The local processor must write this bit to a 0 to allow the 21555 to be configured by the host processor, unless preloaded to 0 by serial ROM. • When 0, the 21555 configuration space can be accessed from both interfaces. • When 1, the 21555 configuration space can only be accessed from the secondary interface. Primary bus accesses, with the exception of the Reset Control Register, receive a target retry. • Reset value is 1 when **pr_ad[3**] is high during reset, 0 when **pr_ad[3]** is low during reset.
11	Secondary Clock Disable	R/W	Secondary clock output disable. (refer to Table 20) • When 0, signal **s_clk_o** is driven as a buffered copy of p_clk. • When 1, signal **s_clk_o** is disabled and driven low. • Reset value is 0 when **pr_ad[5]** is high during primary bus reset; 1 when **pr_ad[5]** is low during primary bus reset.

Table 77. Chip Control 0 Register (Sheet 4 of 4)

This register may be preloaded by serial ROM or programmed by the local processor before host configuration.

- Primary byte offset: CD:CCh
- Secondary byte offset: CD:CCh

Bit	Name	R/W	Description
12	LUT Page Size Extension Bit	R/W	Allows selection of larger page sizes when programming the page size field in the Chip Control 1 configuration register. • When 0, page sizes 256 bytes through 4 MB are available in the page size field. • When 1, page sizes 8 MB through 32 MB are available in the page size field. • Reset value is 0
13	Retry Counter	R/W	Disables or enables all 2^{24} retry counters in both directions. • When 0, all 2^{24} retry counters are enabled. When the 21555 attempts a posted write or delayed transaction as a master and receives 2^{24} target retries, the 21555 discards the transaction. • When 1, all 2^{24} retry counters are disabled and do not place any limits on the number of attempts the 21555 makes when initiating a posted write or delayed transaction. • Reset value is 0
15:14	VGA Mode	R/W	Enables address decoding and transaction forwarding of the following VGA transactions: • Frame buffer memory addresses 000BFFFF:000A0000h • VGA I/O addresses 3BB:3B0h and 3DF:3C0h, where **AD[31:16]** = 0000h and **AD[15:10]** are not decoded. The following values control how the 21555 decodes and forwards VGA memory and I/O transactions: • 00: VGA memory and I/O transactions on the primary and secondary buses are ignored (unless decoded by some other mechanism). • 01: VGA memory and I/O transactions on the primary bus are forwarded to the secondary bus. VGA transactions on the secondary bus are ignored. • 10: VGA memory and I/O transactions on the secondary bus are forwarded to the primary bus. VGA transactions on the primary bus are ignored. • 11: Illegal.The 21555 behavior is unpredictable. Reset value is 00b

Table 78. Chip Control 1 Register (Sheet 1 of 3)

This register may be preloaded by serial ROM or programmed by the local processor before host configuration.

- Primary byte offset: CF:CEh
- Secondary byte offset: CF:CEh

Bit	Name	R/W	Description
0	Primary Posted Write Threshold	R/W	Controls the queue full threshold limit of the downstream posted write queue. When the queue is designated full, the 21555 returns retry to posted writes on the primary bus. Otherwise, the 21555 accepts write data into the posted write queue. • When 0, posted write queue full when less than a cache line is free to post data. • When 1, posted write queue full when less than a half cache line (for CLS=8,16,32) is free to post data. • Reset value is 0b
1	Secondary Posted Write Threshold	R/W	Controls the queue full threshold limit of the upstream posted write queue. When the queue is designated full, the 21555 returns retry to posted writes on the secondary bus. Otherwise, the 21555 accepts write data into the posted write queue. • When 0, posted write queue full when less than a cache line is free to post data. • When 1, posted write queue full when less than a half cache line (for CLS=8,16,32) is free to post data. • Reset value is 0b
3:2	Primary Delayed Read Threshold	R/W	Controls the read data queue threshold for initiating read transactions on the primary bus. When the amount of read data in the queue exceeds the threshold, the 21555 does not initiate a pending upstream delayed memory read transaction on the primary bus. The following values control when the 21555 initiates a memory read: • 00b: At least 8 Dwords free in read data queue for all memory read commands • 01b: Illegal (uses the same behavior as 00b) • 10b: At least one cache line free for MRL and MRM, 8 Dwords free for memory read • 11b: At least one cache line free for all memory read commands **NOTE:** The secondary bus cache line size is used for the threshold calculation. Reset value is 00b
5:4	Secondary Delayed Read Threshold	R/W	Controls the read data queue threshold for initiating read transactions on the secondary bus. When the amount of read data in the queue exceeds the threshold, the 21555 does not initiate a pending downstream delayed memory read transaction on the secondary bus. The following values control when the 21555 initiates a memory read: • 00b: At least 8 Dwords free in read data queue for all memory read commands • 01b: Illegal (uses the same behavior as 00b) • 10b: At least one cache line free for MRL and MRM, 8 Dwords free for memory read • 11b: At least one cache line free for all memory read commands **NOTE:** The primary bus cache line size is used for the threshold calculation. Reset value is 00b

Table 78. Chip Control 1 Register (Sheet 2 of 3)

This register may be preloaded by serial ROM or programmed by the local processor before host configuration.

- Primary byte offset: CF:CEh
- Secondary byte offset: CF:CEh

Bit	Name	R/W	Description
7:6	Subtractive Decode Enable	R/W	Controls subtractive decoding for downstream and upstream I/O transactions. When the 21555 is enabled to perform subtractive decoding in one direction, those transactions are forwarded to the opposite bus with no address translation. Possible values are: • 00: No subtractive decoding is performed on either interface. • 01: I/O subtractive decoding enabled on primary interface. • 10: I/O subtractive decoding enabled on secondary interface. • 11: Illegal. Results are unpredictable. Reset value is 00
11:8	Page Size	R/W	Selects the page size used for the Upstream Memory 2 address range. The total size of this range is dependent on the page size. Possible page size values and their encoding are dependent on the LUT Page Size Extension bit [12] in the Chip Control 0 register. When the LUT Page Size Extension bit is 0, the encodings are: • 0h : Disables the Upstream Memory 2 Base address register. • 1h : 256 bytes • 2h : 512 bytes • 3h : 1 KB • 4h : 2 KB • 5h : 4 KB • 6h : 8 KB • 7h : 16 KB • 8h : 32 KB • 9h : 64 KB • Ah : 128 KB • Bh : 256 KB • Ch : 512 KB • Dh : 1 MB • Eh : 2 MB • Fh : 4 MB When the LUT Page Size Extension bit is 1, the encodings are: • 0h : Disables the Upstream Memory 2 Base address register. • 1h : 8 MB • 2h : 16 MB • 3h : 32 MB • 4h to Fh : Disables the Upstream Memory 2 Base address register. Reset value is 0h

Table 78. Chip Control 1 Register (Sheet 3 of 3)

This register may be preloaded by serial ROM or programmed by the local processor before host configuration.

- Primary byte offset: CF:CEh
- Secondary byte offset: CF:CEh

Bit	Name	R/W	Description
12	I20_ENA	R/W	Enables the I20 message unit. • When 0, the I20 message unit is disabled. Memory accesses to the Inbound and Outbound FIFO registers at CSR offsets 40h and 44h result in **TRDY#** and discarded data on writes, and **TRDY#** with a return of FFFFFFFFh on reads. • When 1, the I20 message unit is enabled. Memory writes cause a posting to the Inbound Post or Outbound Free list; Reads remove an entry from the Inbound Free or Outbound Post list. • Reset value is 0.
15:13	I20_SIZE	R/W	Selects the I20 FIFO size. The 21555 supports the following values: • 000b : 256 entries • 001b : 512 entries • 010b : 1 K entries • 011b : 2 K entries • 100b : 4 K entries • 101b : 8 K entries • 110b : 16 K entries • 111b : 32 K entries Reset value is 000b

Table 79. Chip Status Register

All of the following conditions can cause the assertion of **p_serr_l** or **s_serr_l** if the corresponding **SERR#** enable bit is set and the disable bit for this condition is not set.

- Primary byte offset: D1:D0h
- Secondary byte offset: D1:D0h

Bit	Name	R/W	Description
0	Downstream Delayed Transaction Master Time-out	R/W1TC	This bit is set to a 1 and **p_serr_l** is conditionally asserted when the primary master timeout counter expires and a downstream delayed transaction completion is discarded from the 21555's queues. Reset value is 0
1	Downstream Delayed Read Transaction Discarded	R/W1TC	This bit is set to a 1 and **p_serr_l** is conditionally asserted when the 21555 discards a downstream delayed read transaction request after receiving 2^{24} target retries from the secondary bus target (Retry counters must not be disabled). Reset value is 0
2	Downstream Delayed Write Transaction Discarded	R/W1TC	This bit is set to a 1 and **p_serr_l** is conditionally asserted when the 21555 discards a downstream delayed write transaction request after receiving 2^{24} target retries from the secondary bus target (Retry counters must not be disabled). Reset value is 0

Table 79. Chip Status Register

All of the following conditions can cause the assertion of **p_serr_l** or **s_serr_l** if the corresponding **SERR#** enable bit is set and the disable bit for this condition is not set.

- Primary byte offset: D1:D0h
- Secondary byte offset: D1:D0h

Bit	Name	R/W	Description
3	Downstream Posted Write Data Discarded	R/W1TC	This bit is set to a 1 and **p_serr_l** is conditionally asserted when the 21555 discards a downstream posted write transaction after receiving 2^{24} target retries from the secondary bus target (Retry counters must not be disabled). Reset value is 0
7:4	Reserved	R	Reserved. Returns 0 when read.
8	Upstream Delayed Transaction Master Time-out	R/W1TC	This bit is set to a 1 and **s_serr_l** is conditionally asserted when the secondary master timeout counter expires and an upstream delayed transaction completion is discarded from the 21555's queues. Reset value is 0
9	Upstream Delayed Read Transaction Discarded	R/W1TC	This bit is set to a 1 and **s_serr_l** is conditionally asserted when the 21555 discards an upstream delayed read transaction request after receiving 2^{24} target retries from the primary bus target (Retry counters must not be disabled). Reset value is 0
10	Upstream Delayed Write Transaction Discarded	R/W1TC	This bit is set to a 1 and **s_serr_l** is conditionally asserted when the 21555 discards an upstream delayed write transaction request after receiving 2^{24} target retries from the primary bus target (Retry counters must not be disabled). Reset value is 0
11	Upstream Posted Write Data Discarded	R/W1TC	This bit is set to a 1 and **s_serr_l** is conditionally asserted when the 21555 discards an upstream posted write transaction after receiving 2^{24} target retries from the primary bus target (Retry counters must not be disabled). Reset value is 0
15:12	Reserved	R	Reserved. Returns 0 when read.

Table 80. Generic Own Bits Register

The 21555 implements two generic own bits that can be accessed in either memory or I/O space from either the primary or secondary interface. These bits may be used as an aid to lock resources in software. When a bus master reads the Own bit, it returns 1 if it has already been set, or it returns 0 if the Own bit is available and then automatically sets the bit upon completion of the read. The Own bit is cleared by writing a 1 to the bit. A read-only shadow copy of the bit can be read to check the status of an Own bit without causing the bit to set.

Own bit 0 is bit [0] at CSR offset 0D0h, bits [7:1] are reserved. Own bit 1 is bit [0] at CSR offset 0D1h, bits [7:1] are reserved. Shadow copies of these Own bits may be found at bits [1:0] at CSR offset 0D2h.

These bits may be used as generic own bits, or semaphores. Setting or clearing these bits has no direct hardware effect on other 21555 functions. Byte reads of this location are suggested to avoid unintended side-effects.

- Byte offsets: 0D2:0D0h

Bit	Name	R/W	Description
0	Generic Own Bit 0	ROTS/ W1TC	Generic own bit 0. This bit may be used as a semaphore by either primary or secondary bus masters. When read, current value is returned, and then the bit is automatically set to a 1 if the value read is 0, or kept as a 1 if the value read was 1. Writing a 1 clears this bit to a 0. Writing a 0 has no effect. Generic Own Status 0 can be read to check the state of this bit without side effects. Reset value is 0
7:1	Reserved	R	Reserved. Returns 0 when read.
8	Generic Own Bit 1	ROTS/ W1TC	Generic own bit 1. This bit may be used as a semaphore by either primary or secondary bus masters. When read, current value is returned, and then the bit is automatically set to a 1 if the value read is 0, or kept as a 1 if the value read was 1. Writing a 1 clears this bit to a 0. Writing a 0 has no effect. Generic Own Status 1 can be read to check the state of this bit without side effects. Reset value is 0
15:9	Reserved	R	Reserved. Returns 0 when read.
16	Generic Own 0 Status	R	Returns the current state of Own Bit 0. When checking the state of this bit, the lower two bytes of this Dword location should be masked to prevent unintended side effects.
17	Generic Own 1 Status	R	Returns the current state of Own Bit 1. When checking the state of this bit, the lower two bytes of this Dword location should be masked to prevent unintended side effects.
23:18	Reserved	R	Reserved. Returns 0 when read.

16.6 I2O Registers

This section contains a description of the I2O registers. See Chapter 14 for theory of operation information.

Table 81. I2O Outbound Post_List Status

Byte Offset: 33:30h			
Bit	**Name**	**R/W**	**Description**
2:0	Reserved	R	Reserved. Read only as 0.
3	Outbound Post Status	R	Reflects the status of the Outbound Post_List. • When 0, the Outbound Post_List is empty. The 21555 deasserts **p_inta_l** (unless it is asserted for other reasons). • When 1, the Outbound Post_List is not empty. When the Outbound Post_List Interrupt Mask bit is zero, the 21555 asserts **p_inta_l** as long as this status bit is set. • Reset value is 0
31:4	Reserved	R	Reserved. Read only as 0.

Table 82. I2O Outbound Post_List Interrupt Mask

Byte Offset: 37:34h			
Bit	**Name**	**R/W**	**Description**
2:0	Reserved	R	Reserved. Read only as 0.
3	Outbound Post Mask	R/W	Interrupt mask for Outbound Post_List Status. • When 0, the 21555 asserts **p_inta_l** when the Outbound Post_List Status bit is a 1. • When 1, the 21555 does not assert **p_inta_l** when the Outbound Post_List Status bit is a 1. • Reset value is 1
31:4	Reserved	R	Reserved. Read only as 0.

Table 83. I2O Inbound Post_List Status

Byte Offset: 3B:38h			
Bit	**Name**	**R/W**	**Description**
2:0	Reserved	R	Reserved. Read only as 0.
3	Inbound Post Status	R	Reflects the status of the Inbound Post_List. • When 0, the Inbound Post_List is empty. The 21555 deasserts **s_inta_l** (unless it is asserted for other reasons). • When 1, the Inbound Post_ List is not empty. When the Inbound Post_List Interrupt Mask bit is zero, the 21555 asserts **s_inta_l** as long as this status bit is set. • Reset value is 0
31:4	Reserved	R	Reserved. Read only as 0.

Table 84. I2O Inbound Post_List Interrupt Mask

Byte Offset: 3F:3Ch			
Bit	**Name**	**R/W**	**Description**
2:0	Reserved	R	Reserved. Read only as 0.
3	Inbound Post Mask	R/W	Interrupt mask for Inbound Post_List Status. • When 0, the 21555 asserts **s_inta_l** when the Inbound Post_List Status bit is a 1. • When 1, the 21555 does not assert **s_inta_l** when the Inbound Post_List Status bit is a 1. • Reset value is 1
31:4	Reserved	R	Reserved. Read only as 0.

Table 85. I2O Inbound Queue

Byte Offset: 43:40h			
Bit	**Name**	**R/W**	**Description**
31:0	I2O_IN (P) Reserved (S)	R/(WP)	This register controls the host processor access to the I2O inbound queue. When this register is read from the primary bus, the 21555 returns the value from the head of the I2O inbound Free_List. When this register is written from the primary bus, the 21555 writes the data to the tail of the inbound Post_List. Accesses from the secondary bus are treated as reserved. The actual location of the inbound queue lists are in local memory, and the initial location of the Free_List head and Post_List tail pointers must be programmed by the local processor in the I2O Inbound Free_List Head Pointer and I2O Inbound Post_List Tail Pointer registers.

Table 86. I2O Outbound Queue

Byte offset: 47:44h			
Bit	**Name**	**R/W**	**Description**
31:0	I2O_OUT (P) Reserved (S)	R/(WP)	This register controls the host processor access to the I2O outbound queue. When this register is read from the primary bus, the 21555 returns the value from the head of the I2O outbound Post_List. When this register is written from the primary bus, the 21555 writes the data to the tail of the outbound Free_List. Accesses from the secondary bus are treated as reserved. The actual location of the outbound queue lists are in local memory, and the initial location of the Post_List head and Free_List tail pointers must be programmed by the local processor in the I2O Outbound Post_List Head Pointer and I2O Outbound Free_List Tail Pointer registers.

Table 87. I2O Inbound Free_List Head Pointer

Byte Offsets: 04B:048h			
Bit	**Name**	**R/W**	**Description**
1:0	Reserved	R	Reserved. Returns 0 when read.
31:2	Inbound Free Head Ptr	R/W	Specifies the local memory Dword address of the Inbound Free_List Head Pointer. Increments when the I2O Inbound Queue at offset 40h is read on the primary bus. This pointer automatically wraps when it reaches the upper boundary of the Inbound Free_List.

Table 88. I2O Inbound Post_List Tail Pointer

Byte Offsets: 04F:04Ch			
Bit	**Name**	**R/W**	**Description**
1:0	Reserved	R	Reserved. Returns 0 when read.
31:2	Inbound Post Tail Ptr	R/W	Specifies the local memory Dword address of the Inbound Post_List Tail Pointer. Increments when the I2O Inbound Queue at offset 40h is written on the primary bus. This pointer automatically wraps when it reaches the upper boundary of the Inbound Post_List.

Table 89. I2O Outbound Free_List Tail Pointer

Byte Offsets: 053:050h			
Bit	**Name**	**R/W**	**Description**
1:0	Reserved	R	Reserved. Returns 0 when read.
31:2	Outbound Free Tail Ptr	R/W	Specifies the local memory Dword address of the Outbound Free_List Tail Pointer. Increments when the I2O Outbound Queue at offset 44h is written on the primary bus. This pointer automatically wraps when it reaches the upper boundary of the Outbound Free_List.

Table 90. I2O Outbound Post_List Head Pointer

Byte Offsets: 057:054h			
Bit	**Name**	**R/W**	**Description**
1:0	Reserved	R	Reserved. Returns 0 when read.
31:2	Outbound Post Head Ptr	R/W	Specifies the local memory Dword address of the Outbound Post_List Head Pointer. Increments when the I2O Outbound Queue at offset 44h is read on the primary bus. This pointer automatically wraps when it reaches the upper boundary of the Outbound Post_List.

Table 91. I2O Inbound Post_List Counter

Byte Offsets: 05B:058h			
Bit	**Name**	**R/W**	**Description**
15:0	Inbound Post Ctr	R/(WS)	When read, returns the number of entries in the Inbound Post_List. Decrements by 1 when this location is written from the secondary interface with any data value if bit [31] of this register is written with a 0 during the same write. When bit [31] is written with a 1, the 21555 loads the counter with the value written. Increments when the Inbound Queue at offset 40h is written from the primary interface. Reset value : 0
30:16	Reserved	R	Reserved. Read only as 0.
31	LD_IPC	W1TL(S)	Load Inbound Post_List Counter. • When written with a 1 at the same time as the Inbound Post Car bits [15:0], it loads the Inbound Post_List Counter with the value on **s_ad[15:0]** during that same write. • When written with a 0, or if **s_cbe_l[3]** is 1, decrements the Inbound Post_List Counter. Reads always return 0.

Table 92. I2O Inbound Free_List Counter

Byte Offsets: 05F:05Ch.			
Bit	**Name**	**R/W**	**Description**
15:0	Inbound Free Ctr	R/(WS)	When read, returns the number of entries in the Inbound Free_List. Increments by 1 when this location is written from the secondary interface with any data value if bit [31] of this register is written with a 0 during the same write. When bit [31] is written with a 1, the 21555 loads the counter with the value written. Decrements when the Inbound Queue at offset 40h is read from the primary interface, except when the counter is zero. The 21555 does not decrement when the counter is 0. Reset value is 0
30:16	Reserved	R	Reserved. Read only as 0.
31	LD_IFC	W1TL(S)	Load Inbound Free_List Counter. • When written with a 1 at the same time as the Inbound Free Ctr bits [15:0], loads the Inbound Free_List Counter with the value on **s_ad[15:0]** during that same write. • When written with a 0, or if **s_cbe_l[3]** is 1, increments the Inbound Free_List Counter. Reads always return 0.

Table 93. I2O Outbound Post_List Counter

Byte Offsets: 063:060h			
Bit	**Name**	**R/W**	**Description**
15:0	Outbound Post Ctr	R/(WS)	When read, returns the number of entries in the Outbound Post_List. Increments by 1 when this location is written from the secondary interface with any data value if bit [31] of this register is written with a 0 during the same write. When bit [31] is written with a 1, the 21555 loads the counter with the value written. Decrements when the Outbound Queue at offset 44h is read from the primary interface, except when the counter is zero. The 21555 does not decrement when the counter is 0. Reset value is 0
30:16	Reserved	R	Reserved. Read only as 0.
31	LD_OPC	W1TL(S)	Load Outbound Post_List Counter. When written with a 1 at the same time as the Outbound Post Ctr bits [15:0], it loads the Outbound Post_List Counter with the value on **s_ad[15:0]** during that same write. When written with a 0, or if **s_cbe_l[3]** is 1, it increments the Outbound Post_List Counter. Reads always return 0.

Table 94. I2O Outbound Free_List Counter

Byte Offsets: 067:064h			
Bit	**Name**	**R/W**	**Description**
15:0	Outbound Free Ctr	R/(WS)	When read, returns the number of entries in the Outbound Free_List. Decrements by 1 when this location is written from the secondary interface with any data value when bit [31] of this register is written with a 0 during the same write. When bit [31] is written with a 1, the 21555 loads the counter with the value written. Increments when the Outbound Queue at offset 44h is written from the primary interface. Reset value is 0
30:16	Reserved	R	Reserved. Read only as 0.
31	LD_OPC	W1TL(S)	Load Outbound Free_List Counter. • When written with a 1 at the same time as the Outbound Free Ctr bits [15:0], it loads the Outbound Free_List Counter with the value on **s_ad[15:0]** during that same write. • When written with a 0, or if **s_cbe_l[3]** is 1, it decrements the Outbound Free_List Counter. Reads always return 0.

16.7 Interrupt Registers

This section contains information about interrupt registers. See Chapter 11 for theory of operation information.

Table 95. Chip Status CSR

Byte Offsets: 083:082h			
Bit	**Name**	**R/W**	**Description**
0	PM_D0	R/W1TC	Power Management Transition to D0. The 21555 sets this bit when it is transitioned from a low power D1 or D2 state to a high power D0 state. When the corresponding Chip IRQ Mask bit for this event is a 0, the 21555 asserts **s_inta_l** to indicate to the subsystem that it is being brought to a higher power state. Writing a 1 clears this bit to a 0. Writing a 0 has no effect. Reset value is 0
1	Subsystem Event	R/W1TC	Generic subsystem event bit. The 21555 sets this bit when a deasserting (rising) edge is detected on **s_pme_l**. When **s_pme_l** is not used for power management purposes, it may be used to signal some other subsystem event. When the Chip IRQ Mask bit for this event is a 0, the 21555 asserts **p_inta_l** to indicate to the host system that this signal was deasserted. Writing a 1 clears this bit to a 0. Writing a 0 has not effect. Reset value is 0
15:2	Reserved	R	Reserved. Returns 0 when read.

Table 96. Chip Set IRQ Mask Register

Byte Offsets: 085:084h			
Bit	**Name**	**R/W**	**Description**
0	Set_D0M	R/W1TS	• When 0, signal **s_inta_l** is asserted on the 21555's secondary interface when the corresponding chip event bit is a 1, indicating a return of power state to D0. • When 1, the corresponding chip event bit does not generate an interrupt. Writing a 1 to a bit in this register sets the Chip IRQ Mask bit to 1. Writing a 0 to any bit in this register has no effect. Reading this register returns the current status of the Chip IRQ Mask bits. • Reset value is 1
1	Set_Sstat	R/W1TS	• When 0, signal **p_inta_l** is asserted on the 21555's primary interface when the corresponding chip event bit is a 1, indicating a deasserting edge on **s_pme_l**. • When 1, the corresponding chip event bit does not generate an interrupt. Writing a 1 to a bit in this register sets the Chip IRQ Mask bit to 1. Writing a 0 to any bit in this register has no effect. Reading this register returns the current status of the Chip IRQ Mask bits. • Reset value is 1
15:2	Reserved	R	Reserved. Returns 0 when read.

Table 97. Chip Clear IRQ Mask Register

Byte Offsets: 087:086h			
Bit	**Name**	**R/W**	**Description**
0	Clr_D0M	R/W1TC	• When 0, signal **s_inta_l** is asserted on the 21555's secondary interface when the corresponding chip event bit is a 1, indicating a return of power state to D0. • When 1, the corresponding chip event bit does not generate an interrupt. Writing a 1 to a bit in this register clears the Chip IRQ Mask bit to 0. Writing a 0 to any bit in this register has no effect. Reading this register returns the current status of the Chip IRQ Mask bits. • Reset value is 1
1	Clr_Sstat	R/W1TC	• When 0, **p_inta_l** is asserted on the 21555's primary interface when the corresponding chip event bit, indicating a deasserting edge on **s_pme_l**, is a 1. • When 1, the corresponding chip event bit does not generate an interrupt. Writing a 1 to a bit in this register clears the Chip IRQ Mask bit to 0. Writing a 0 to any bit in this register has no effect. Reading this register returns the current status of the Chip IRQ Mask bits. • Reset value is 1
15:2	Reserved	R	Reserved. Returns 0 when read.

Table 98. Upstream Page Boundary IRQ 0 Register

Byte Offset: 08B:088h			
Bit	**Name**	**R/W**	**Description**
31:0	PAGE0_IRQ	R/W1TC	Each bit in this register corresponds to a page entry in the lower half of the Upstream Memory 2 range. Bit 0 corresponds to the first (lowest order) page, and bit 31 corresponds to the 32nd page. The 21555 sets the appropriate bit when it successfully transfers data to/from the initiator that addresses the last Dword in a page. When the Upstream Page Boundary 0 IRQ Mask bit corresponding to that page is zero, the 21555 asserts **s_inta_l**. Reset value is 0

Table 99. Upstream Page Boundary IRQ 1 Register

Byte Offset: 08F:08Ch			
Bit	**Name**	**R/W**	**Description**
31:0	PAGE1_IRQ	R/W1TC	Each bit in this register corresponds to a page entry in the upper half of the Upstream Memory 2 range. Bit 0 corresponds to the 33rd page, and bit 31 corresponds to the 64th (highest order) page. The 21555 sets the appropriate bit when it successfully transfers data to/ from the initiator that addresses the last Dword in a page. When the Upstream Page Boundary 1 IRQ Mask bit corresponding to that page is zero, the 21555 asserts **s_inta_l**. Reset value is 0

Table 100. Upstream Page Boundary IRQ Mask 0 Register

Byte Offset: 093:090h			
Bit	**Name**	**R/W**	**Description**
31:0	PAGE0_MASK	R/W	• When 0, the 21555 asserts **s_inta_l** when the corresponding status bit in the Upstream Page Boundary IRQ 0 register is set. • When 1, the 21555 does not assert **s_inta_l** when the corresponding status bit in the Upstream Page Boundary IRQ 0 register is set. • Reset value is FFFFFFFFh

Table 101. Upstream Page Boundary IRQ Mask 1 Register

Byte Offset: 097:094h			
Bit	**Name**	**R/W**	**Description**
31:0	PAGE1_MASK	R/W	• When 0, the 21555 asserts **s_inta_l** when the corresponding status bit in the Upstream Page Boundary IRQ 1 register is set. • When 1, the 21555 does not assert **s_inta_l** when the corresponding status bit in the Upstream Page Boundary IRQ 1 register is set. • Reset value is FFFFFFFFh

Table 102. Primary Clear IRQ and Secondary Clear IRQ Registers

These registers affect primary and secondary interrupts in the same way and are described together.

	Primary Clear IRQ	Secondary Clear IRQ
Byte Offset:	099:098h	09B:09Ah

Bit	Name	R/W	Description
15:0	CLR_IRQ	R/W1TC	This register controls the state of the Primary or Secondary Interrupt Request bits. • When 0, Does not cause the corresponding primary or secondary interrupt signal to be asserted. • When 1, the primary or secondary interrupt signal is asserted when the corresponding IRQ Mask bit is zero. Writing a 1 to a bit in this register clears the corresponding interrupt request bit to 0. Writing a 0 to any bit in this register has no effect. Reading this register returns the current status of the interrupt request bits. • Reset value is 0

Table 103. Primary Set IRQ and Secondary Set IRQ Registers

These registers affect primary and secondary interrupts in the same way and are described together.

Offsets	Primary Set IRQ	Secondary Set IRQ
Byte	09D:09Ch	09F:09Eh

Bit	Name	R/W	Description
15:0	SET_IRQ	R/W1TS	This register controls the state of the Primary or Secondary Interrupt Request bits. • When 0, Does not cause the corresponding primary or secondary interrupt signal to be asserted. • When 1, the primary or secondary interrupt signal is asserted if the corresponding IRQ Mask bit is zero. Writing a 1 to a bit in this register sets the corresponding interrupt request bit to 1. Writing a 0 to any bit in this register has no effect. Reading this register returns the current status of the interrupt request bits. • Reset value is 0

Table 104. Primary Clear IRQ Mask and Secondary Clear IRQ Mask Registers

These registers affect primary and secondary interrupts in the same way and are described

	Primary Clear IRQ Mask	Secondary Clear IRQ Mask
Byte Offset:	0A1:0A0h	0A3:0A2h

Bit	Name	R/W	Description
15:0	CLR_IRQM	R/W1TC	• When 0, an interrupt is generated on the 21555's primary or secondary interface when the corresponding Primary or Secondary Interrupt Request bit is a 1. • When 1, the corresponding interrupt request bit cannot generate an interrupt. Writing a 1 to a bit in this register clears the IRQ Mask bit to 0. Writing a 0 to any bit in this register has no effect. Reading this register returns the current status of the IRQ Mask bits. • Reset value is FFFFFFFFh

Table 105. Primary Set IRQ Mask and Secondary Set IRQ Mask Registers

These registers affect primary and secondary interrupts in the same way and are described together.

Offsets	Primary Set IRQ Mask	Secondary Set IRQ Mask
Byte	0A5:0A4h	0A7:0A6h

Bit	Name	R/W	Description
15:0	SET_IRQM	R/W1TS	• When 0, an interrupt is generated on the 21555's primary or secondary interface when the corresponding Primary or Secondary Interrupt Request bit is a 1. • When 1, the corresponding interrupt request bit cannot generate an interrupt. Writing a 1 to a bit in this register sets the IRQ Mask bit to 1. Writing a 0 to any bit in this register has no effect. Reading this register returns the current status of the IRQ Mask bits. • Reset value is FFFFFFFFh

16.8 Scratchpad Registers

See Chapter 11 for theory of operation information.

Table 106. Scratchpad 0 Through Scratchpad 7 Registers (Sheet 1 of 2)

Bit	Name	R/W	Byte Offset:	Description
31:0	SCRATCH0	R/W	0AB:0A8h	32-bit scratchpad register 0.
31:0	SCRATCH1	R/W	0AF:0ACh	32-bit scratchpad register 1.
31:0	SCRATCH2	R/W	0B3:0B0h	32-bit scratchpad register 2.

Table 106. Scratchpad 0 Through Scratchpad 7 Registers (Sheet 2 of 2)

Bit	Name	R/W	Byte Offset:	Description
31:0	SCRATCH3	R/W	0B7:0B4h	32-bit scratchpad register 3.
31:0	SCRATCH4	R/W	0BB:0B8h	32-bit scratchpad register 4.
31:0	SCRATCH5	R/W	0BF:0BCh	32-bit scratchpad register 5.
31:0	SCRATCH6	R/W	0C3:0C0h	32-bit scratchpad register 6.
31:0	SCRATCH7	R/W	0C7:0C4h	32-bit scratchpad register 7.

16.9 PROM Registers

This section describes the six PROM registers. See Chapter 8 for theory of operation information.

Table 107. Primary Expansion ROM BAR

This register defines an address range in which a memory read transaction on the primary interface of the 21555 results in a read access to the PROM interface. The Primary Expansion ROM Setup register controls the size of the address range requested by the Primary Expansion ROM Base Address register. The Primary Expansion ROM Setup register must be loaded either from the serial ROM or by the local processor before configuration software running on the host processor can access this register. Local processor access should be completed before the Configuration Lockout flag is cleared.

- Primary byte offset: 33:30h
- Secondary byte offset: 73:70h

Bit	Name	R/W	Description
0	Address Decode Enable	R/W	Enables the 21555 to respond to accesses to its expansion ROM space. When this BAR is disabled, this bit will return zero when read. • When 1, the 21555 responds to memory accesses to expansion ROM space when the Memory Enable bit is also set. • When 0, the 21555 does not respond to accesses directed to this address space. • Reset value is 0.
11:1	Reserved	R	Reserved. Returns 0 when read.
31:12	Base Address	R/W	These bits are used to indicate the size of the expansion ROM space and to set the base address of the range. Bits [23:11] of the Primary Expansion ROM Setup register determine the function of the corresponding bit in this register. • When a bit in the Primary Expansion ROM Setup register is 0, the same bit in this register is a read-only bit and always returns 0 when read. • When a bit in the Primary Expansion ROM Setup register is 1, the same bit in this register is writable and returns the value last written when read. • When this BAR is enabled, bits [31:24] are always writable. Writing a zero to bit [24] of the Primary Expansion ROM Setup register disables this BAR. The minimum size for this address range is 4 KB. The maximum size is 16 MB. • Reset value is 0 (disabled).

.

Table 108. Primary Expansion ROM Setup Register

This register may be preloaded by serial ROM or programmed by the local processor before host configuration.

- Primary byte offset: C3:C0h
- Secondary byte offset: C3:C0h

Bit	Name	R/W	Description
11:0	Reserved	R	Reserved. Read only as 0.
23:12	Size	R/(WS)	These bits specify the size of the address range requested by the Primary Expansion ROM Base Address register. • When a bit is 1, the corresponding bit in the Primary Expansion ROM Base Address register functions as a readable and writable bit. • When a bit is 0, the corresponding bit in the Primary Expansion ROM Base Address register functions as a read-only bit that always returns zero when read. • Reset value is 0 (disabled).
24	BAR_Enable	R/(WS)	The location of this bit in the preload sequence, as listed in “Serial Preload Sequence” on page 16-180. • When 0, the Primary Expansion ROM BAR is disabled and reads as 0. • When 1, the Primary Expansion ROM BAR is enabled, with size specified by this setup register. • Reset value is 0
31:25	Reserved	R	Reserved. Returns 0 when read.

Table 109. ROM Setup Register

Byte Offsets: 0C9:0C8h			
Bit	**Name**	**R/W**	**Description**
1:0	Access Time	R/W	Number of **p_clk** cycles that **pr_cs_l** asserts low (in default mode) or **pr_ale_l** drives high (in multiple device mode) for a PROM or other external device access. Possible values: • 00: 8 times 33 MHz **p_clk** cycle time 00:16 times 66 MHz **p_clk** cycle time • 01: 16 times 33 MHz **p_clk** cycle time 01: 32 times 66 MHz **p_clk** cycle time • 10: 64 times 33 MHz **p_clk** cycle time 10:128 times 66 MHz **p_clk** cycle time • 11: 256 times 33 MHz **p_clk** cycle time 11:512 times 66 MHz **p_clk** cycle time A 33 MHz **p_clk** is specified by **p_m66ena** low, and a 66 MHz **p_clk** is specified by **p_m66ena** high. Reset value is 00b
7:2	Reserved	R	Reserved. Reads only as 0.
15:8	Strobe Mask	R/W	Read and write strobe timing mask. This 8-bit field defines the setup time, duration, and hold time of **pr_rd_l** and **pr_wr_l** during the device select assertion. A 1 asserts the strobe, a 0 deasserts it. The LSB is the first bit in time, the MSB is the last bit. Each bit is one eighth of the access time, or for the following access time values: • 00: 1 each 33 MHz or 2 each 66 MHz **p_clk** cycles per bit • 01: 2 each 33 MHz or 4 each 66 MHz **p_clk** cycles per bit • 10: 8 each 33 MHz or 16 each 66 MHz **p_clk** cycles per bit • 11: 32 each 33 MHz or 64 each 66 MHz **p_clk** cycles per bit A 33 MHz **p_clk** is specified by **p_m66ena** low, and a 66 MHz **p_clk** is specified by **p_m66ena** high. Reset value is 01111110b

Table 110. ROM Data Register

Byte Offsets: 0CAh			
Bit	**Name**	**R/W**	**Description**
7:0	ROM_DATA	R/W	• When the PROM Start bit is set, contains the read or write data for bits [7:0] of the PROM. • When the Serial ROM Start bit is set, contains the read or write data for bits [7:0] of the serial ROM.

Table 111. ROM Address Register

Byte Offsets: 0CE:0CCh			
Bit	**Name**	**R/W**	**Description**
23:0	ROM_ADDR	R/W	Contains the byte address of the PROM read or write access used when the PROM Start bit is set to a 1. Contains the byte address and Opcode used when the Serial ROM Start bit is set to a 1. The byte address is contained on bits [8:0]. The opcode is contained on bits [10:9]. Possible opcode values are: • 00: write all, erase all, write enable, programming disable • 01: write • 10: read • 11: erase Reset value is 000400h (serial ROM read at address 0)

Table 112. ROM Control Register (Sheet 1 of 2)

Byte Offsets: 0CFh			
Bit	**Name**	**R/W**	**Description**
0	Serial ROM Start/Busy	R/W1TS	Starts a serial ROM read, write, or polling operation and returns the completion status of the access. When written with a 1, performs the serial ROM operation indicated by the serial ROM opcode. This bit is automatically cleared by the 21555 when the serial ROM access is complete. This bit should not be written unless both the Serial ROM Start and PROM Start bits are 0. Writing a 0 to this bit has no effect. When the previous serial ROM operation was a write all, erase all, write, or erase, writing this bit causes the 21555 to poll the serial ROM to test for the completion of the operation. The result of the poll operation is reflected in bit 3 of this register. Reset value is 0
1	PROM Start/Busy	R/W1TS	Starts a PROM read or write operation and returns the completion status of the access. When written with a 1, the 21555 performs the PROM operation indicated by the PROM Read/Write Control bit. This bit is automatically set when the 21555 performs a PROM read from the Primary Expansion ROM address space. This bit is automatically cleared by the 21555 when the PROM access is complete. This bit should not be set unless both the Serial ROM Start and PROM Start bits are 0. Writing a 0 to this bit has no effect. Reset value is 0

Table 112. ROM Control Register (Sheet 2 of 2)

Byte Offsets: 0CFh			
Bit	**Name**	**R/W**	**Description**
2	Read/Write Control	R/W	PROM read/write control bit. This bit may be written with the same CSR access that sets the PROM Start bit. • When 0, the 21555 performs a read of the PROM when the PROM Start bit is set to a 1. • When 1, the 21555 performs a write of the PROM when the PROM Start bit is set to a 1. • Reset value is 0
3	SROM_POLL	R	This bit reflects the status of the serial ROM as a result of a polling operation following a write all, erase all, write, or erase operation. This bit is set automatically by the 21555 when one of these operations is initiated, and cleared when a subsequent poll of the serial ROM indicates that the operation is complete. Reset value is 0
7:4	Reserved	R	Reserved. Reads only as 0.

16.10 SROM Registers

This sections describes the SROM registers. See Chapter 9 for theory of operation information.

Table 113. Mode Setting Configuration Register (Sheet 1 of 2)

This register reflects the various mode settings selected by strapping the **pr_ad** pins, as well as whether the 64-bit extension is enabled. • Primary byte offset: D6h • Secondary byte offset: D6h			
Bit	**Name**	**R/W**	**Description**
0	Serial Preload Enabled	R	Indicates whether a serial preload was performed. • When 0, the serial preload enable sequence was not detected and the register preload was not performed. • When 1, the serial preload enable sequence was detected and the register preload was performed.
1	Primary Lockout Reset Value	R	Indicates the primary lockout reset value determined by sampling **pr_ad[3]** during reset. • When 0, signal **pr_ad[3]** was sampled low, causing the this bit to be low upon completion of chip reset. • When 1, signal **pr_ad[3]** was sampled high, causing the this bit to be set high upon completion of chip reset.
2	Synchronous Enable	R	Indicates whether synchronous or asynchronous mode was selected by sampling **pr_ad[4]** during reset. • When 0, signal **pr_ad[4]** was sampled low, selecting synchronous mode. • When 1, signal **pr_ad[4]** was sampled high, selecting asynchronous mode.

Table 113. Mode Setting Configuration Register (Sheet 2 of 2)

This register reflects the various mode settings selected by strapping the **pr_ad** pins, as well as whether the 64-bit extension is enabled.

- Primary byte offset: D6h
- Secondary byte offset: D6h

Bit	Name	R/W	Description
3	**s_clk_o** Enable	R	Indicates whether **s_clk_o** is enabled, determined by sampling **pr_ad[5]** during reset. • When 0, signal **pr_ad[5]** was sampled low, causing **s_clk_o** to be disabled. • When 1, signal **pr_ad[5]** was sampled high, causing **s_clk_o** to be enabled.
4	Secondary Central Function Enable	R	Indicates whether secondary bus central functions are enabled, determined by sampling **pr_ad[6]** during reset. • When 0, signal **pr_ad[6]** was sampled low, causing secondary central functions to be enabled. • When 1, signal **pr_ad[6]** was sampled high, causing secondary central functions to be disabled.
5	Arbiter Enable	R	Indicates whether the secondary bus arbiter is enabled, determined by sampling **pr_ad[7]** during reset. • When 0, signal **pr_ad[7]** was sampled low, causing the secondary bus arbiter to be disabled. • When 1, signal **pr_ad[7]** was sampled high, causing the secondary bus arbiter to be enabled.
6	Primary 64-Bit Extension	R	Indicates whether the primary bus 64-bit extension is enabled. • When 0, the primary bus 64-bit extension is disabled. • When 1, the primary bus 64-bit extension is enabled.
7	Secondary 64-Bit Extension	R	Indicates whether the secondary bus 64-bit extension is enabled. • When 0, the secondary bus 64-bit extension is disabled. • When 1, the secondary bus 64-bit extension is enabled.

Table 114. Serial Preload Sequence (Sheet 1 of 3)

Not all of the bits in the sequence are used. Bits that are not used must be 0 (zero)

Byte offset	Description
00h	Register preload control • Bits [7:6] must read as 10b to enable the register preload; otherwise, the serial ROM read is terminated and registers remain at their reset values. • Bits [5:0] are reserved and must be 0.
01h	00000000b (Reserved)
02h	00000000b (Reserved)
03h	00000000b (Reserved)
04h	Primary Programming Interface
05h	Primary Sub-Class Code
06h	Primary Base Class Code
07h	Subsystem Vendor ID [7:0]
08h	Subsystem Vendor ID [15:8]

Table 114. Serial Preload Sequence (Sheet 2 of 3)

Not all of the bits in the sequence are used. Bits that are not used must be 0 (zero)	
Byte offset	**Description**
09h	Subsystem ID [7:0]
0Ah	Subsystem ID [15:8]
0Bh	Primary Minimum Grant
0Ch	Primary Maximum Latency
0Dh	Secondary Programming Interface
0Eh	Secondary Sub-Class Code
0Fh	Secondary Base Class Code
10h	Secondary Minimum Grant
11h	Secondary Maximum Latency
12h	Downstream Memory 0 Setup [7:0]. Bits [0, 7:4] are not loaded and should be 0.
13h	Downstream Memory 0 Setup [15:8]. Bits [11:8] are not loaded and should be 0.
14h	Downstream Memory 0 Setup [23:16]
15h	Downstream Memory 0 Setup [31:24]
16h	Downstream I/O or Memory 1 Setup [7:0]. Bits [5:4] are not loaded and should be 0.
17h	Downstream I/O or Memory 1 Setup [15:8]
18h	Downstream I/O or Memory 1 Setup [23:16]
19h	Downstream I/O or Memory 1 Setup [31:24]
1Ah	Downstream Memory 2 Setup [7:0]. Bits [0, 7:4] are not loaded and should be 0.
1Bh	Downstream Memory 2 Setup [15:8]. Bits [11:8] are not loaded and should be 0.
1Ch	Downstream Memory 2 Setup [23:16]
1Dh	Downstream Memory 2 Setup [31:24]
1Eh	Downstream Memory 3 Setup [7:0]. Bits [0, 7:4] are not loaded and should be 0.
1Fh	Downstream Memory 3 Setup [15:8]. Bits [11:8] are not loaded and should be 0.
20h	Downstream Memory 3 Setup [23:16]
21h	Downstream Memory 3 Setup [31:24]
22h	Downstream Memory 3 Setup Upper 32 Bits [7:0]
23h	Downstream Memory 3 Setup Upper 32 Bits [15:8]
24h	Downstream Memory 3 Setup Upper 32 Bits [23:16]
25h	Downstream Memory 3 Setup Upper 32 Bits [31:24]
26h	• Bit [0]: Primary Expansion ROM Setup [24] (enable) • Bits [3:1]: Not loaded. Should be 0. • Bits [7:4]: Primary Expansion ROM Setup [15:11]
27h	Primary Expansion ROM Setup [23:16]
28h	Upstream I/O or Memory 0 Setup [7:0]. Bits [5:4] are not loaded and should be 0.
29h	Upstream I/O or Memory 0 Setup [15:8]
2Ah	Upstream I/O or Memory 0 Setup [23:16]

Table 114. Serial Preload Sequence (Sheet 3 of 3)

Not all of the bits in the sequence are used. Bits that are not used must be 0 (zero)	
Byte offset	**Description**
2Bh	Upstream I/O or Memory 0 Setup [31:24]
2Ch	Upstream Memory 1 Setup [7:0]. Bits [0, 7:4] are not loaded and should be 0.
2Dh	Upstream Memory 1 Setup [15:8]. Bits [11:8] are not loaded and should be 0.
2Eh	Upstream Memory 1 Setup [23:16]
2Fh	Upstream Memory 1 Setup [31:24]
30h	Chip Control 0 [7:0]
31h	Chip Control 0 [15:8]. Bits [13:12] are not loaded and should be 0.
32h	Chip Control 1 [7:0]
33h	Chip Control 1 [15:8]
34h	Arbiter Control [7:0]
35h	Arbiter Control [15:7]. Bits [15:10] are not loaded and should be 0.
36h	Primary **SERR#** Disable. Bit [7] is not loaded and should be 0.
37h	Secondary **SERR#** Disable. it [7] is not loaded and should be 0.
38h	Power Management Data 0
39h	Power Management Data 1
3Ah	Power Management Data 2
3Bh	Power Management Data 3
3Ch	Power Management Data 4
3Dh	Power Management Data 5
3Eh	Power Management Data 6
3Fh	Power Management Data 7
40h	Reserved
41h	• [1:0] 00b (Reserved) • [2] BiST Supported • [3] Power Management Data Register Enable • [5:4] Power Management Control and Status [14:13] • [7:6] Power Management Capabilities Register [1:0]
42h	• [0] Power Management Capabilities Register [2] • [1] Power Management Capabilities Register [5] • [7:2] Power Management Capabilities Register [14:9]

16.11 Arbiter Control

This chapter describes the arbitration control registers. See Chapter 10 for theory of operation information.

Table 115. Arbiter Control Register

This register may be preloaded by serial ROM or programmed by local processor before host configuration. • Primary byte offset: D3:D2h • Secondary byte offset: D3:D2h			
Bit	**Name**	**R/W**	**Description**
9:0	Arbiter Control	R/W	Each bit controls whether a secondary bus master is assigned to the high priority arbiter ring or the low priority arbiter ring. Bits [8:0] correspond to request inputs **s_req_l[8:0]**, respectively. Bit [9] corresponds to the internal 21555 secondary bus request. • When 0, indicates that the master belongs to the low priority group. • When 1, indicates that the master belongs to the high priority group. • Reset value is 10 0000 0000b.
10	Bus Parking Control	R/W	Controls whether the 21555 parks on itself or on the last master to use the bus. • When 0, during bus idle, the 21555 parks the bus on the last master to use the bus. • When 1, during bus idle, the 21555 parks the bus on itself. The bus grant is removed from the last master and internally asserted to the 21555. • Reset value is 0b.
15:11	Reserved	R	Reserved. Returns 0 when read.

16.12 Error Registers

This section describes the primary and secondary SERR# disable registers. See Chapter 12 for theory of operation information.

Table 116. Primary SERR# Disable Register

This register may be preloaded by serial ROM or programmed by the local processor before host configuration. This register controls the ability of the 21555 to assert **p_serr_l** for a particular condition. When the bit is a 0, the assertion of **p_serr_l** is not masked for this event.
When the bit is a 1, the assertion of **p_serr_l** is masked for this event.

- Primary byte offset: D4h
- Secondary byte offset: D4h

Bit	Name	R/W	Description
0	Downstream Delayed Transaction Master Time-out	R/W	Disables **p_serr_l** assertion when a downstream master time-out condition is detected and the downstream transaction is discarded. Reset value is 0
1	Downstream Delayed Read Transaction Discarded	R/W	Disables **p_serr_l** assertion when 21555 discards a downstream delayed read transaction request after receiving 2^{24} target retries from secondary bus target. Reset value is 0
2	Downstream Delayed Write Transaction Discarded	R/W	Disables **p_serr_l** assertion when 21555 discards a downstream delayed write transaction request after receiving 2^{24} target retries from secondary bus target. Reset value is 0
3	Downstream Posted Write Data Discarded	R/W	Disables **p_serr_l** assertion when 21555 discards a downstream posted write transaction after receiving 2^{24} target retries from secondary bus target. Reset value is 0
4	Target Abort during Downstream Posted Write	R/W	Disables **p_serr_l** assertion when 21555 detects a target abort on the secondary interface in response to a downstream posted write. Reset value is 0
5	Master Abort during Downstream Posted Write	R/W	Disables **p_serr_l** assertion when the 21555 detects a master abort on the secondary interface when initiating a downstream posted write. Reset value is 0
6	Downstream Posted Write Parity Error	R/W	Disables **p_serr_l** assertion when the 21555 detects **s_perr_l** asserted during a downstream posted write. Reset value is 0
7	Reserved	R	Reserved. Returns 0 when read.

Table 117. Secondary SERR# Disable Register

This register may be preloaded by serial ROM or programmed by the local processor before host configuration. This register controls the ability of the 21555 to assert **s_serr_l** for a particular condition. When the bit is a 0, the assertion of **s_serr_l** is not masked for this event.
When the bit is a 1, the assertion of **s_serr_l** is masked for this event.

- Primary byte offset: D5h
- Secondary byte offset: D5h

Bit	Name	R/W	Description
0	Upstream Delayed Transaction Master Timeout	R/W	Disables **s_serr_l** assertion when an upstream master timeout condition is detected and the upstream transaction is discarded. Reset value is 0

Table 117. Secondary SERR# Disable Register

1	Upstream Delayed Read Transaction Discarded	R/W	Disables **s_serr_l** assertion when the 21555 discards an upstream delayed read transaction request after receiving 2^{24} target retries from the primary bus target. Reset value is 0
2	Upstream Delayed Write Transaction Discarded	R/W	Disables **s_serr_l** assertion when the 21555 discards an upstream delayed write transaction request after receiving 2^{24} target retries from the primary bus target. Reset value is 0
3	Upstream Posted Write Data Discarded	R/W	Disables **s_serr_l** assertion when the 21555 discards an upstream posted write transaction after receiving 2^{24} target retries from the primary bus target. Reset value is 0
4	Target Abort during Upstream Posted Write	R/W	Disables **s_serr_l** assertion when the 21555 detects a target abort on the primary interface in response to an upstream posted write. Reset value is 0
5	Master Abort during Upstream Posted Write	R/W	Disables **s_serr_l** assertion when the 21555 detects a master abort on the primary interface when initiating an upstream posted write. Reset value is 0
6	Upstream Posted Write Parity Error	R/W	Disables **s_serr_l** assertion when the 21555 detects **p_perr_l** asserted during an upstream posted write. Reset value is 0
7	Reserved	R	Reserved. Returns 0 when read.

16.13 Init Registers

This section describes the Power management, Reset, and Hot-swap registers. See Chapter 2 for theory of operation information.

Table 118. Power Management ECP ID and Next Pointer Register

- Primary byte offset: DD:DCh
- Secondary byte offset: DD:DCh

Bit	Name	R/W	Description
7:0	PM ECP ID	R	Power Management Enhanced Capabilities Port ID. Read only as 01h to identify these ECP registers as Power Management registers.
15:8	PM Next Ptr	R	Pointer to next ECP registers. Reads as E4 to indicate the first register of the next set of ECP registers, which support Vital Product Data, and resides at offset E4h.

Table 119. Power Management Capabilities Register

Bits [14:9,5,2:0] are loadable through the serial ROM or are programmable by the local processor.
- Primary byte offset: DF:DEh
- Secondary byte offset: DF:DEh

Bit	Name	R/W	Description
2:0	PM Version	R/(WS)	Power Management Version. Loadable by serial ROM. Reset value is Signal 001b to indicate that this device is compliant to the *PCI Power Management Interface Specification, Revision 1.1*.
3	PME Clock	R	Clock Required for **PME#** Assertion. Reads as 1 to indicate that a clock is required to assert **PME#**, when any of bits [15:11] in this register are asserted. Read as 0 when bits [15:11] are all 0, indicating that the 21555 does not assert **PME#**.
4	APS	R	Auxiliary Power Source. Not defined since the 21555 does not have PME# support from $D3_{cold}$. Read only as 0.
5	DSI	R/(WS)	Device-Specific Initialization. Loadable by serial ROM. • When 0, indicates that the 21555 does not have device-specific initialization requirements. • When 1, indicates that the 21555 has device-specific initialization requirements. • Reset value is 0
8:6	Reserved	R	Reserved. Read only as 0.
9	D1 Support	R/(WS)	D1 Power State Support Indicator. Loadable by serial ROM. • When 0, indicates that the 21555 does not support the D1 power management state. • When 1, indicates that the 21555 supports the D1 power management state. • Reset value is 0
10	D2 Support	R/(WS)	D2 Power State Support Indicator. Loadable by serial ROM. • When 0, indicates that the 21555 does not support the D2 power management state. • When 1, indicates that the 21555 supports the D2 power management state. • Reset value is 0
15:11	PME Support	R/(WS)	PME# support. Indicates whether the 21555 asserts **p_pme_l** when in a given power state. Bit 11 corresponds to D0; bit 15 corresponds to $D3_{cold}$. Bits [14:11] are loadable by serial ROM. Bit 15 always reads as 0 and is not loaded by serial ROM nor writable from the secondary interface, since the 21555 never asserts **PME#** when in $D3_{cold}$. Reset value is 0 to indicate that the 21555 does not implement the **PME#** pin.

.

Table 120. Power Management Control and Status Register

Bits [14:13] are loadable by serial ROM or are programmable by the local processor.
- Primary byte offset: E1:E0h
- Secondary byte offset: E1:E0h

Bit	Name	R/W	Description
1:0	PWR State	R/W	Power State. Reflects the current power state of the 21555. When an unimplemented power state is written to this register, the 21555 completes the write transaction, ignores the write data, and does not change the value of this field. D0 and D3 are always implemented. Support of D1 and D2 is determined by serial ROM preload. 00b : D0 (required) 01b : D1 (optional) 10b : D2 (optional) 11b : D3 (required) Reset value is 00b
3:2	Reserved	R	Reserved. Read only as 00b.
4	DYN DATA	R	Dynamic Data. Reads as 0 to indicate that the 21555 does not support dynamic data reporting.
7:5	Reserved	R	Reserved. Read only as 000b.
8	PME_EN	R/W	PME# Enable. Read only as 0 when the PME Support bits are all 0. Otherwise, this is a read/write bit. • When 0, the 21555 will not assert **p_pme_l**. • When 1, the 21555 is enabled to assert **p_pme_l**. • Reset value is 0
12:9	DATA_SEL	R/W	Data Select. This register is enabled by loading a “1” for the PM Data Enable function in the serial ROM. See “Serial Preload Sequence” on page 16-180. For values of 7:0, this register selects one of eight bytes of data loaded by serial ROM to be placed in the power management data register. For values of 15:8, a 0 is returned in the data register. • When not enabled, this register always returns 0 when read. • Reset value is 000b.
14:13	Data Scale	R/(WS)	Data Scale. Indicates the scaling factor of the value in the power management data register. Loadable by serial ROM. Reset value is 00b
15	PME Status	R/W1TC	PME Status. The 21555 sets this bit to a 1 when **s_pme_l** is asserted and the PME# Support bit for the current power state is a 1. This corresponds to when the 21555 would normally assert **p_pme_l**, but regardless of the state of the PME_En bit. Writing a 1 clears this bit. Writing a 0 has no effect. Reset value is 0.

Table 121. PMCSR Bridge Support Extensions

- Primary byte offset: E2h
- Secondary byte offset: E2h

Bit	Name	R/W	Description
7:0	BSE	R	Bridge Support Extensions. Read only as 00h.

Table 122. Power Management Data Register

• Primary byte offset: E3h • Secondary byte offset: E3h			
Bit	**Name**	**R/W**	**Description**
7:0	PM Data	R	Power Management Data register. Reflects one of eight bytes loaded by serial ROM, or reads as 0. Bytes are selected by the data select register. Reset value is 00h

Table 123. Reset Control Register

This register is accessible from the primary interface regardless of the state of the Primary Lockout Reset Value bit • Primary byte offset: DB:D8h • Secondary byte offset: DB:D8h			
Bit	**Name**	**R/W**	**Description**
0	Secondary Reset	R/(WP)	Secondary bus reset. • When 0, the 21555 deasserts **s_rst_l**. This bit must be cleared by a configuration write when it is set by a configuration write. Otherwise, it clears automatically after 100 µs or when **p_rst_l** deasserts. • When 1, the 21555 asserts **s_rst_l**. This bit is set automatically when the Chip Reset bit is written with a 1 or when **p_rst_l** is asserted, or is set with a configuration write. • Reset value is 0 (disabled).
1	Chip Reset	R/(WP)	Chip reset control. • When 1, the 21555 performs a chip reset and to assert **s_rst_l**. Data buffers, configuration registers, and both the primary and secondary interfaces are reset to their initial state. The 21555 clears this bit once chip reset is complete. • Reset value is 0
2	Subsystem Status	R	Reflects the state of the **s_pme_l** pin. When not used for power management, the **s_pme_l** pin may be used to indicate local subsystem status. • When 0, signal **s_pme_l** is at a deasserted high level. • When 1, signal **s_pme_l** is at an asserted low level.
3	l_stat Status	R	Reflects the state of the l_stat pin. • When 0, signal **l_stat** is low. • When 1, signal **l_stat** is high.
31:3	Reserved	R	Reserved. Reads only as 0.

Table 124. CompactPCI Hot-Swap Capability Identifier and Next Pointer Register

Offsets	HS ECP ID	HS Next Pointer
Primary byte	ECh	EDh
Secondary byte	ECh	EDh

Bit	Name	R/W	Description
7:0	HS ECP ID	R	Enhanced capabilities ID. Reads only as 06h to indicate that these are CompactPCI Hot-Swap registers.
7:0	HS NXT PTR	R	Pointer to next set of ECP registers. Reads only as 0 to indicate that these are the last ECP registers in this list.

Table 125. CompactPCI Hot-Swap Control Register (Sheet 1 of 2)

- Primary byte offset: EF:EEh
- Secondary byte offset: EF:EEh

Bit	Name	R/W	Description
0	Reserved	R	Reserved. Read only as 0.
1	ENUM_MASK	R/W	ENUM# Interrupt Mask. • When 0, the 21555 asserts **p_enum_l** when an insertion or removal event occurs. • When 1, the 21555 does not assert **p_enum_l**. • Reset value is 0
2	Reserved	R	Reserved. Read only as 0.
3	LED On/Off (LOO)	R/W	LED On/Off (LOO) Control. Allows software control of the **l_stat** pin and therefore the state of the LED. • When 0, the 21555 tristates **l_stat**. When REM STAT is low, the LED is off when the ejector handle is closed and on when the ejector handle is open. When REM STAT is high, **l_stat** is not tristated but continues to be driven by the 21555 (LED is off). • When 1, the 21555 drives **l_stat** high and the LED is forced on. • Reset value is 0

Table 125. CompactPCI Hot-Swap Control Register (Sheet 2 of 2)

• Primary byte offset: EF:EEh • Secondary byte offset: EF:EEh			
Bit	**Name**	**R/W**	**Description**
5:4	Reserved	R	Returns 0 when read.
6	REM STAT	R/ W1TC	Signal **p_enum_l** Removal Status. The 21555 sets this bit to a 1 when **l_stat** is sampled high and **p_rst_l** is deasserted, signaling an impending removal. This bit is cleared when software writes a 1. Clearing this bit causes the 21555 to tristate **l_stat**. Writing a 0 has no effect. • When 1, the 21555 is asserting **p_enum_l** to indicate that the card is about to be removed. • Reset value is 0
7	INS STAT	R/ W1TC	Signal **p_enum_l** Insertion Status. The 21555 sets this bit to a 1 when **l_stat** is sampled low (ejector handle is closed), the serial preload is complete and the Primary Lockout Reset Value bit is cleared, indicating that the card is ready for host initialization. This bit is cleared when software writes a 1. Writing a 0 has no effect. • When 1, the 21555 is asserting **p_enum_l** to indicate that the card has just been inserted. • Reset value is 0

16.14 JTAG Registers

This chapter presents the theory of operation information about the 21555 JTAG registers. See Chapter 13 for theory of operation information.

Table 126. JTAG Instruction Register Options (Sheet 1 of 2)

The 4-bit instruction register selects the test mode and features. The instruction codes are shown in **Table 126**. These instructions select and control the operation of the boundary-scan and bypass registers. The instruction register is loaded through the **tdi** pin. The instruction register has a serial shift-in stage from which the instruction is then loaded in parallel.

Instruction Register Contents	Instruction Name	Test Register Selected	Operation
0000	EXTEST	Boundary-scan	External test (drives pins from boundary-scan register).
0001	SAMPLE	Boundary-scan	Samples inputs.
0010	HIGHZ	Bypass	Tristates all output and I/O pins except the tdo pin.

† Manufacturer's identification number: 0000 0001 0011
Design part number: 1001 0010 0110 0010
Version: 0000

Table 126. JTAG Instruction Register Options (Sheet 2 of 2)

The 4-bit instruction register selects the test mode and features. The instruction codes are shown in Table 126. These instructions select and control the operation of the boundary-scan and bypass registers. The instruction register is loaded through the **tdi** pin. The instruction register has a serial shift-in stage from which the instruction is then loaded in parallel.			
Instruction Register Contents	**Instruction Name**	**Test Register Selected**	**Operation**
0011	CLAMP	Bypass	Drives pins from the boundary-scan register and selects the bypass register for shifts.
0100	IDCODE†	Idcode	Reads the manufacturer's identification number, the design part number, and the design version number.
0101-1111	BYPASS	Bypass	Selects the bypass register for shifts.

† Manufacturer's identification number: 0000 0001 0011
Design part number: 1001 0010 0110 0010
Version: 0000

Table 127. Bypass Register

The bypass register is a 1-bit shift register that provides a means for effectively bypassing the JTAG test logic through a single-bit serial connection through the device from **tdi** to **tdo**. At board-level testing, this helps reduce the overall length of the scan ring.

Table 128. Boundary-Scan Register

The boundary-scan register (BSR) is a single-shift register-based path formed by boundary-scan cells placed at the device's signal pins. The register is accessed through the **tdi** and **tdo** pins of the JTAG port. Each boundary-scan cell operates in conjunction with the current instruction and the current state in the test access port controller state machine. The function of the BSR cells is determined by the Input, Output, and Bidirectional pins.	
Pin	**Description**
Input-only pins	The boundary-scan cell is basically a 1-bit shift register. The cell supports sample and shift operations.
Output-only pins	The boundary-scan cell comprises a 1-bit shift register and an output multiplexer. The cell supports the sample, shift, and drive output functions.
Bidirectional pins:	The boundary-scan cell is identical to the output-only pin cell, but it captures test data from the incoming data line. The cell supports sample, shift, drive output, and hold output functions.

Table 129. Boundary Scan Order

TBD table lists the boundary-scan register order and the group disable controls. The group disable control either enables or tristates its corresponding group of bi-directional drivers. When the value of a group disable control bit is 0, the output driver is enabled. When the value is 1, the driver is tri-stated. There are **TBD** groups of bi-directional drivers, and therefore **TBD** group disable control bits.

The group disable number column in **TBD** shows which group disable bit controls the corresponding output driver. Group disable bits do not affect input-only pins, so those pins have a blank rather than a group number in that column. The group disable control can control pins on either side of where the group disable boundary-scan register cell is placed. The group disable boundary-scan register cells have a boundary-scan register order number entry, but do not have either a corresponding pin number or signal name.

Data shifts from **tdi** into the most significant bit of the boundary-scan register out to **tdo**.

Table to be included - **TBD**.

16.15 *VPD* Registers

This section provides a description of the VPD registers. See Chapter 15 for theory of operation information.

Table 130. Vital Product Data (VPD) ECP ID and Next Pointer Register

• Primary byte offset: E5:E4h • Secondary byte offset: E5:E4h			
Bit	**Name**	**R/W**	**Description**
7:0	VPD ECP ID	R	VPD Enhanced Capabilities Port ID. Read only as 03h to identify these ECP registers as VPD registers.
15:8	VPD Next Ptr	R	Pointer to next ECP registers. Reads as ECh to point to the next set of ECP registers, supporting CompactPCI Hot-Swap.

Table 131. Vital Product Data (VPD) Address Register

- Primary byte offset: E7:E6h
- Secondary byte offset: E7:E6h

Bit	Name	R/W	Description
8:0	VPD Addr	R/W	Vital Product Data Address. Contains the VPD byte address of the serial ROM location to be accessed. Valid VPD byte addresses are 17F: 000h. VPD starts at base address 080h in the serial ROM. The VPD byte address contained in this register is added to the VPD base address to obtain the final serial ROM address.
14:9	Reserved	R	Reserved. Read Only as 0.
15	VPD Flag	R/W	VPD Flag. Starts a VPD serial ROM access and indicates completion of the operation. • *When written with a 0:* A 4-byte serial ROM read is performed starting at the VPD location indicated by bits [8:0]. Note that this operation is not necessarily Dword aligned. When the read is complete, the 21555 sets this bit to a 1. • *When written with a 1:* A 4-byte serial ROM write is performed starting at the VPD location indicated by bits [8:0]. Note that this operation is not necessarily Dword aligned. • When the write is complete, the 21555 sets this bit to a 0.

Table 132. VPD Data Register

- Primary byte offset: EB:E8h
- Secondary byte offset: EB:E8h

Bit	Name	R/W	Description
31:0	VPD Data	R/W	VPD Data. Contains the VPD read or write data. For a read, this register should be read after a read operation was initiated and the 21555 has returned the VPD Flag bit to a 1. For a write, this register should be written with the write data before the operation is initiated with a write to the VPD Address and VPD Flag bits. VPD read and write operations are always 4-byte operations. Byte 0 contains the data corresponding to the starting VPD byte address. Byte 1, 2, and 3 contain successive bytes. Note that Byte 0 is not necessarily Dword aligned.

Acronyms A

- 1D – One-dimensional
- 2D – Two-dimensional
- AGP – Accelerated Graphics Port
- ANSI – American National Standards Institute
- API – Application Programming Interface
- BAR – Base Address Register
- BiST – Built-In Self-Test
- CLS – Cache Line Sizes
- CSR – Control and Status Registers
- DAC – Dual Address Cycle
- DRC – Delayed Read Completion
- DRR – Delayed Read Request
- DWC – Delayed Write Completion
- DWR – Delayed Write Request
- ETSI – European Telecommunications Standards Institute
- FFT – Fast Fourier transform
- FIR – Finite impulse response
- Flash – Nonvolatile memory
- GOB – Group of blocks
- GPIO – General Purpose Input Output
- I20 – Intelligent Input/Output
- IDCT – Inverse discrete cosine transform
- IEC – International Electrotechnical Commission
- IIR – Infinite impulse response
- IOP – Input Output Processor
- IPP – Integrated Performance Primitives
- ISO – International Standards Organization
- ITU – International Telecommunication Union
- IXA – Internet Exchange Architecture; for example: Intel® IXA.
- LMS – Least mean square
- MB – Macroblock
- MC – Motion compensation

- MFAs – Message Frame Addresses
- MPEG – Moving Pictures Experts Group
- MV – Motion vector
- MVC – Part Control Number
- MWI – Memory Write and Invalidate
- OBMC – Overlapped block motion compensation
- OS – Operating system
- PPB – PCI-to-PCI bridge
- PQFP – Plastic Quad Flat Package
- quiesced – The PCI card is no longer performing or scheduling transactions.
- RAM – Read only memory
- ROM – Random access memory
- SA – StrongARM or StrongARM Architecture
- SAC – Single Address Cycle
- Sfs – Saturated fixed scale
- TAP – (JTAG) Test Access Port
- Vdd _vio or V/IO – (S)scondary or (P)rimary_Voltage Input or Output. In PCI specifications it is defined as V/IO
- VGA – Video Graphics Adapter
- VLD – Variable length decoding
- Vss – Voltage for Substrate & Sources, usually ground potential.

Index

Q

R

S

T

U

V

W