SG2010
PCI–to–StarFabric Bridge

Hardware Reference Manual

Revision Information: Revision 1.2 19 October 2004

StarGen, Inc., 225 Cedar Hill Street, Suite 22, Marlborough, MA 01752
www.stargen.com

October 2004

StarGen, Inc. believes the information in this publication is correct; however, the information is subject to change without notice.

StarGen, Inc. does not claim that the use of its products in the manner described in this publication will not infringe on any existing or future patent rights, nor do the descriptions contained in this publication imply the granting of licenses to make, use, or sell equipment or software in accordance with the description.

©StarGen, Inc. 2000, 2004. All rights reserved.
Printed in U.S.A.

StarGen, StarFabric, and the STARGEN logo are trademarks of StarGen, Inc.

All other trademarks and registered trademarks are the property of their respective owners.

Preface

1 Introduction

2 Feature Summary

3 Operation

List of Figures

List of Tables

Preface

This manual describes the StarGen SG2010 PCI-to-StarFabric Bridge device.

Audience

This manual is written for a technical audience using or evaluating the SG2010.

Overview

This manual contains the following chapters and appendices, a glossary, and an index:

Chapter 1	Introduction – Overview of the SG2010
Chapter 2	Feature List
Chapter 3	Operation – Describes the functional operation of the SG2010.
Chapter 4	Registers – Describes all of SG2010's registers and their mappings.
Chapter 5	Signal Pin Descriptions – Defines the signals on SG2010's pins.
Chapter 6	Signal Pin List – Lists all the signal pins and their BGA locations

Conventions

Register Access Abbreviations

Access	Definition
R	Read only
RTC	Read to clear
RTS	Read to set
RTDEC	Read to decrement
RTINC	Read to increment
R/W	Read, write
R/W1TC	Read, write 1 to clear
R/W1TS	Read, write 1 to set
WRZ	Write, read returns zero

Associated Documents

StarFabric Architecture Specification

*SG1010 StarFabric Switch Hardware Reference Manua*l

Revision History

Revision Number	Date yy/mm/dd	Description
1.0	**02/01/ 2003**	First Release.
1.1	**10/18/ 2004**	Change operating Temperature to Industrial Specification -40C to 85C Move Electrical Specification to product datasheet Move signal internal termination section to product datasheet.
1.2	**10/19/ 2004**	Removed the following statement from the RSVD pin description “ Should be tied low through a weak pulldown resistor.“ These pins should be left as no connects.

1

Introduction

StarGen's PCI-StarFabric Bridge (SG2010) device interfaces 64-bit or 32-bit PCI buses operating at 66MHz or 33MHz to StarFabric, a universal switch fabric.The SG2010 translates PCI traffic into serial frame format for transmission across the switch fabric.By connecting the SG2010's serial interfaces to other bridges or to StarGen's StarFabric switch devices, flexible topologies can be designed to fit specific application requirements for bandwidth, reliability, and a number of endpoints or slots.

The SG2010 is a multi-function device.The 'Bridge' function supports legacy address-routed traffic which provides 100% compatibility with existing PCI software including configuration, BIOS, OS and drivers. The 'Gateway' function provides fabric-native path and multicast routing capability and other enhanced features.The PCI interface is selected as a primary or secondary PCI interface by an external pin. This same pin selects whether the SG2010 is a root or leaf node in the StarFabric.

The StarFabric interface consists of two 2.5Gbps full duplex links providing 2.5Gbps bandwidth in both directions simultaneously. Four aggregated 622Mbps LVDS differential pairs comprise each link. The two links can be bundled to create a 5Gbps full duplex link to another StarFabric device or can be used separately as redundant connections.

Example applications for the StarFabric using the SG2010 are:

- Multi-Service Access Platforms
- DSLAMs-DSL Access Multiplexers
- Voice Over IP Gateways (VOIP)
- Edge Routers/Switches
- Wireless Basestations
- Computer Telephony Integration platforms

STARGEN 2

Feature Summary

The SG2010 has the following features:

- Scalability and Performance
  - 2 StarFabric interface links, 2.5Gbps, full duplex each link
  - Links can be bundled to create 5.0Gbps, full duplex point-to-point connection
  - Supports either an 8-channel 43-bit local address space or a 50-bit global address space
- Compatibility
  - Support for three routing methods: standard PCI addressing (address routing), path routing, and multi-cast routing
  - Standard PCI addressing supports 100% PCI software compatibility
  - Compliant with the PCI Local Bus Specification Revision 2.2, the PCI-to-PCI Bridge Architecture Specification Revision 1.1, and the CompactPCI Hot Swap Specification
  - Physical layer interface is compliant with the IEEE 1596.3 and TIA/EIA-644 Low-Voltage Differential Signaling (LVDS) standards
- Quality of Service
  - Credit based flow control: path-based (next-turn) credits and class-of-service (CoS) credits
  - 4 classes of service: asynchronous (uses address-routing credits), isochronous, multicast, and provisioning
- Reliability, Availability, Serviceability
  - Link-by-link CRC checking on all traffic
  - Fault detection and isolation
  - Redundant path routing capability, optional automatic fail-over
  - Path protection capability for secure operation
  - Hot-pluggable links

  - CompactPCI Hot Swap support

- Additional Features
  - Supports operation as a root or leaf in a StarFabric
  - Flexible event dispatch and handling, either remotely or locally
  - In-band PCI interrupt routing support through the switch fabric
  - Supports software generated PCI transactions
  - Supports software generated StarFabric frames
  - Supports eight semaphores with six operations per semaphore
  - SROM & Flash ROM interfaces for register preload and power-up configuration
  - Eight LED indicators - either under software control or reflects differential pair status
  - PCI bus arbiter supporting up to 9 bus masters
  - Up to eight general purpose I/O pins

3

Operation

3.1 Addressing

3.1.1 Addressing Models

The SG2010 supports two addressing models – a StarFabric addressing model and a PCI addressing model. To support these two addressing models, the SG2010 implements two major functions – a PCI-to-PCI bridge (Bridge) function and a Gateway PCI-to-StarFabric function. The Bridge function supports the PCI addressing model within the fabric and the Gateway function performs translations between the PCI and StarFabric addressing models.

The Bridge function can be disabled in the SG2010, but the Gateway function is always present.

3.1.1.1 StarFabric Addressing Model

The StarFabric addressing model uses a path, a channel, and an offset. The Gateway is the only function in the SG2010 that uses the StarFabric addressing model. The Bridge adheres solely to the PCI addressing model and does not understand StarFabric addresses. The Gateway translates between PCI addresses and StarFabric addresses using the Segment, Path, and Source and Destination Channel Tables. For more information, see Section 3.1.1.2.

StarFabric protocol designates destination Channel 255 for register mapping in StarFabric address space. The Gateway implements a standard set of StarFabric registers, called the StarFabric Component (SFC) Header. These registers are accessible through Channel 255 starting at offset 0. The Gateway's CSRs are also accessible through Channel 255 starting at the 16KB offset boundary (offset 4000h).

Bridge and Gateway PCI configuration registers are dual-mapped into Channel 255 address space.

3.1.1.2 PCI Addressing Model

The Bridge comprises part of a PCI hierarchy in the fabric and performs translations between PCI transactions and address-routed frames. These are protocol translations, not address translations.

Addressing

In the PCI addressing model, the Gateway appears as a PCI device on the PCI bus.

3.1.1.2.1 PCI Addressing Model of the Bridge Function

The Bridge is modeled as a standard transparent PCI-to-PCI Bridge. It implements a Type1 (PCI-to-PCI bridge) configuration header. With the Bridge function, the SG2010 can function with only standard PCI plug-and-play initialization code. The Bridge is configured the same as any transparent PCI-to-PCI bridge and can forward PCI configuration, I/O, and memory transactions to downstream devices through the fabric.

The SG2010 Root pin indicates whether the SG2010 is a root or a leaf in the fabric, and consequently, whether the PCI bus is a primary or secondary PCI bus. If the SG2010 is a root, the PCI bus is a primary bus, downstream transactions flow from PCI to the fabric, and upstream transactions flow from the fabric to PCI. The Bridge's configuration registers are accessible from PCI using a Type0 configuration transaction, but are not accessible from the StarFabric interface.

If the SG2010 is a leaf, the PCI bus is a secondary bus. Downstream transactions flow from the fabric to PCI bus, and upstream transactions flow from the PCI bus to the fabric. The Bridge's configuration registers are accessible only from the StarFabric interface if a PCI frame is received indicating a Type0 configuration operation.

The Bridge converts between PCI transactions and address-routed frames, but the address model of both is the same. Nodes direct address-routed frames through the fabric by decoding the frame address against a set of address ranges. The address ranges are defined at each node by the PCI-to-PCI bridge base and limit address registers and related control bits.

3.1.1.2.2 PCI Addressing Model of the Gateway Function

The Gateway is modeled as a PCI device. It has one addressable PCI interface, which always faces the PCI bus (as opposed to the Bridge, which has two addressable PCI interfaces, one of which is physically a StarFabric interface). The Gateway implements a Type0 configuration register header for PCI configuration.

The Gateway translates between PCI transactions and path-routed/multicast frames. A path-routed frame is assigned a set of turns at the origin and is directed to the terminus through the fabric based on these turns – no further address decodes are needed beyond the initial PCI address decode at the origin. Multicast-routed frames are assigned an ID at the origin and are routed through the fabric based on table lookup at each node.

To use the Gateway to translate and forward PCI transactions into the fabric, additional software must initialize the tables and registers required for this translation. These tables and registers are located in the Gateway CSRs, which are accessible by PCI transactions through the Gateway BAR0 and BAR1 using memory and I/O transactions, respectively.

3.1.1.3 SG2010 Functional Modes

The SG2010 can be used in one of three basic modes:

- Root mode, Bridge function is enabled
- Leaf mode, Bridge function is enabled
- Gateway-only mode

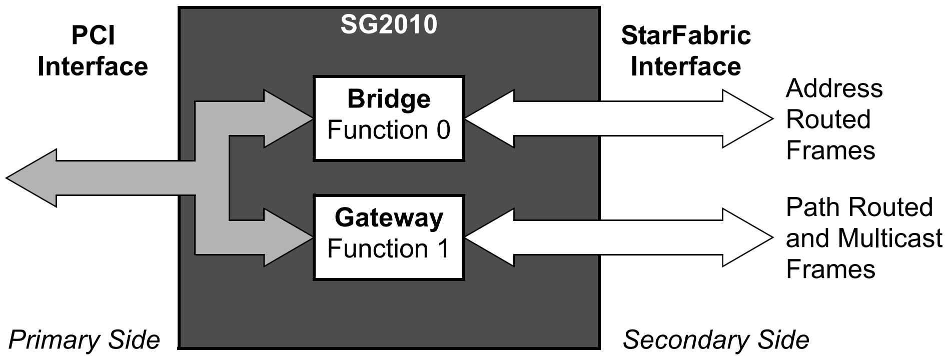
The SG2010 can also be designated as a root or a leaf in Gateway-only mode. However, in terms of frame/transaction forwarding, root/leaf is not distinguished in Gateway-only mode, since transactions and frames are translated and forwarded identically. This is not true when the Bridge is enabled.

3.1.1.3.1 Root mode with Bridge Enabled

When the SG2010 is a root, the PCI interface is connected to the primary bus and the fabric interface represents the secondary bus. This mode is also referred to as multi-function mode.

Figure 3–1 shows a block diagram of the SG2010 configuration as a root with its Bridge function enabled. This figure shows the types of traffic supported on each interface.

Figure 3–1 SG2010 as a Root



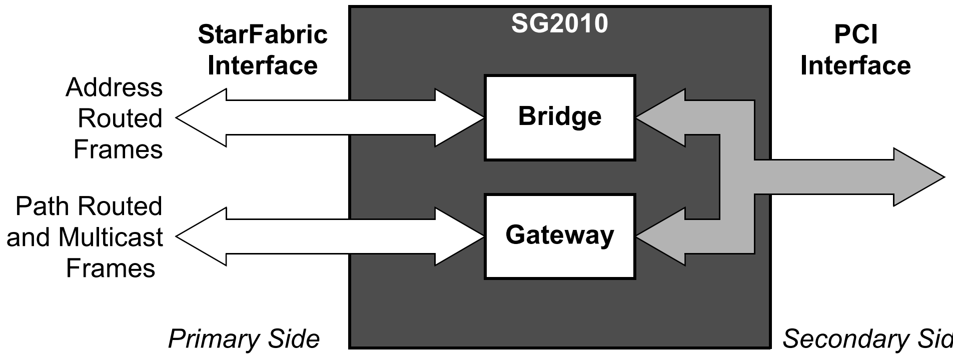
In this mode, the Gateway and the Bridge form a multifunction device. The configuration space of both functions is accessed from the PCI bus using a Type0 configuration transaction, but a single IDSEL signal is used. Configuration Function0 specifies and access to the Bridge function and Function1 specifies an access to the Gateway function.

The root is responsible for initiating fabric enumeration (regardless of whether the Bridge is enabled). Fabric enumeration is important in the PCI addressing model as it identifies which links in the fabric are branches in the PCI hierarchical tree. The root is considered to be the most upstream bridge in the fabric's PCI hierarchy – all PCI configuration initiates from the PCI bus connected to the root.

3.1.1.3.2 SG2010 as a Leaf

When the SG2010 is a leaf, the PCI interface is connected to the secondary bus and one of the ports on the fabric interface is the primary bus. Figure 3–2 shows the SG2010 configuration when it is a leaf.

Figure 3–2 SG2010 as a Leaf



In this mode, the Gateway is logically represented as a separate PCI device located on the Bridge's secondary bus. That is, it is in the same level of hierarchy as the secondary bus devices. This mode is also referred to as secondary subordinate mode.

If the two links comprise two different ports, one port is assigned to be the root port and the other port is not considered to be part of the PCI hierarchy. The root port is assigned during fabric enumeration.

By default, the Bridge is fully transparent. Every PCI device downstream of the Bridge, including the Gateway if the SG2010 is a leaf, is fully visible to the host and their resources are mapped into the global PCI memory map.

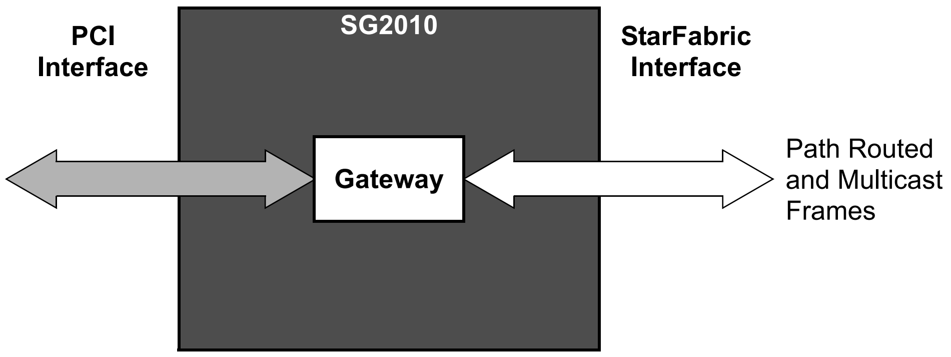
A local PCI subsystem may have resources that it wishes to hide from the host, either because a local processor manages these resources, or because the resources consume a large space in the global address map. When the SG2010 is a leaf, three mechanisms are provided to control access to and from secondary bus devices:

- Hide Gateway address windows BAR2 through BAR5 from the host
- Hide from the host any or all secondary bus devices, including the Gateway, through an IDSEL mask
- Block upstream memory forwarding inside or outside of a programmable window

3.1.1.3.3 SG2010 as a Gateway-only Device

In the third configuration mode, the Bridge is disabled and only the Gateway is present. Figure 3–3 shows the SG2010 in Gateway-only mode.

Figure 3–3 SG2010 as a Gateway-only Device



In Gateway-only mode, the Gateway is visible for PCI configuration only from the PCI bus. Since the Bridge function is required to create a PCI hierarchy in the fabric, using the Gateway-only mode at the root prevents a PCI address-routed hierarchy from being constructed, and isolates the entire fabric from the root's PCI bus. Using the Gateway-only mode at a leaf isolates a PCI subsystem from the PCI host.

The only way to forward PCI transactions in Gateway-only mode is to translate between PCI transactions and path-routed or multicast frames. If the SG2010 is in gateway-only mode and receives an address-routed frame, it drops the frame, signals an Address Routing Failure event, and if a response frame is required, returns a Software Routing Failure failure type.

3.1.1.3.4 SG2010 Functional Mode Summary

Table 3–1 shows a summary of the SG2010 functional modes and the properties of each mode.

Table 3–1 SG2010 Functional Modes

Mode	PCI Configuration	Notes
Root (Multifunction)	PCI is primary; StarFabric is secondary. Bridge and Gateway provide a multifunction configuration interface to the host.	Initiates fabric enumeration.
Leaf (Secondary subordinate)	PCI is secondary; StarFabric is primary. Bridge and Gateway provide a hierarchical configuration model to the host. Gateway is a PCI device on the secondary bus of the Bridge. Enhanced addressing modes can be enabled on the Bridge.	Gateway must be able to respond to address-routed frames from the fabric addressing BAR0 or BAR1 (CSRs). Gateway BARs have three modes of visibility to the host: • All BARs visible • Only BAR0 and 1 visible • Gateway not visible
Gateway-only	Provides no address-routing support into or out of the fabric. Provides private local addressing support.	All frames translated are path-routed/multicast frames.

Table 3–2 summarizes the functional differences between the SG2010 as a root and as a leaf.

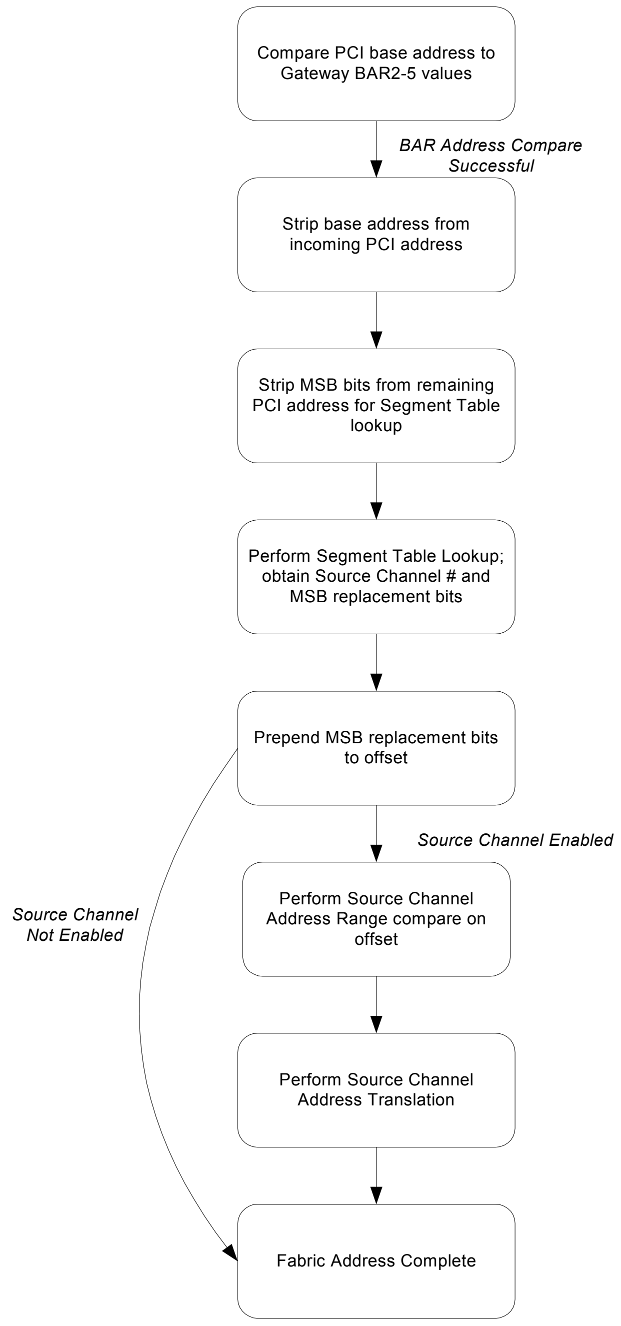
Table 3–2 Root and Leaf Functional Differences

Function	Root Behavior	Leaf Behavior
PCI device model	Single multi-function device (if Bridge enabled)	PCI device behind a P2P device (if Bridge enabled)
Address decoding	PCI traffic positively decoded (Bridge)	PCI traffic inversely decoded (Bridge)
	StarFabric address-routed frames inversely decoded	StarFabric address-routed frames not decoded
	Bridge configuration registers not accessible from link	Bridge configuration registers not accessible from PCI
	Supports all configuration Type 1 transaction forwarding from PCI (if Bridge enabled)	Supports only Type 1 configuration write forwarding for translation to special cycle from PCI (if Bridge enabled)
	Type 1 to Type 0 conversion to Star-Fabric interface only (if Bridge enabled)	Type 1 to Type 0 conversion to PCI interface only (if Bridge enabled)
	Secondary bus address blocking not available	Secondary bus address blocking can be enabled if Bridge is enabled
Initialization	Initiates fabric enumeration	Does not initiate fabric enumeration
	Generates a maskable reset on RST# and LRST# assertion	Does not generate maskable reset on RST# and LRST# assertion
	All Port Map Table updates through snooping, if Bridge enabled	Port Map Table update for bridge-only configuration through channel 255 provisioning write frame (if Bridge enabled)
Events	Input event signals are masked	Input event signals create signal event frames (if Bridge enabled)
	If Event Table Enable not set, all events are handled locally (at root)	If Event Table Enable not set, all events are handled remotely (at root)

3.1.2 Address Decoding in the Fabric Addressing Model

This section describes how PCI transactions are decoded for translation into path-routed and multicast addresses, and how path-routed frame address offsets are translated back into PCI addresses. Figure 3–4 summarizes this translation.

Figure 3–4 PCI to Fabric Address Translation


Compare PCI base address to Gateway BAR2-5 values
BAR Address Compare Successful
Strip base address from incoming PCI address
Strip MSB bits from remaining PCI address for Segment Table lookup
Perform Segment Table Lookup; obtain Source Channel # and MSB replacement bits
Prepend MSB replacement bits to offset
Source Channel Enabled
Source Channel Not Enabled
Perform Source Channel Address Range compare on offset
Perform Source Channel Address Translation
Fabric Address Complete

3.1.2.1 PCI to Fabric Address Translation

The SG2010 performs the following steps when receiving a PCI transaction and translating it to a path-routed frame:

- PCI address decode
- Segment Table look-up
- Path Table look-up
- Optional source channel address translation and other source channel operations

The following sections describe each step.

3.1.2.1.1 PCI Transaction Decoding

The SG2010 Gateway base address registers BAR2 through BAR5 define up to four address ranges for accepting PCI transactions and translating them to path-routed or multicast frames. Each BAR may be individually enabled for address decoding. These BARs can be set up in 32-bit and 64-bit configurations as shown in Table 3–3. A 64-bit BAR uses two BAR registers and can comprise only BAR2 and BAR3 for one 64-bit BAR, or BAR4 and BAR5 for a second 64-bit BAR.

Table 3–3 Gateway BAR2 - BAR5 Allowable Configurations

Configuration	BAR2	BAR3	BAR4	BAR5
One 32-bit range	Enabled	Disabled	Disabled	Disabled
Two 32-bit ranges	Enabled	Enabled	Disabled	Disabled
Three 32-bit ranges	Enabled	Enabled	Enabled	Disabled
Four 32-bit ranges	Enabled	Enabled	Enabled	Enabled
One 64-bit range	Enabled: 64-bit enable set	Enabled: upper 32 bits of BAR2	Disabled	Disabled
Two 64-bit ranges	Enabled: 64-bit enable set	Enabled: upper 32 bits of BAR2	Enabled: 64-bit enable set	Enabled: upper 32 bits of BAR4
One 64-bit range and one 32-bit range	Enabled: 64-bit enable set	Enabled: upper 32 bits of BAR2	Enabled	Disabled
One 64-bit range and wo 32-bit ranges	Enabled: 64-bit enable set	Enabled: upper 32 bits of BAR2	Enabled	Enabled

The size of each address range can be configured. The 32/64-bit configuration and BAR sizes are specified using the corresponding BAR Setup registers (described in Sections 4.8.7.1 and 4.8.7.2) in Gateway device-specific configuration space. The setup registers must be initialized before PCI configuration takes place. These setup registers are dual-mapped into CSR space, and can also be written through serial ROM preload.

When the SG2010 detects a PCI transaction with an address matching one of the base addresses, the SG2010 responds with DEVSEL_L. This transaction is accepted and translated into a path-routed or multicast frame. The following sections describe this translation.

Note that the SG2010 does not respond to incoming frames from the fabric containing an offset that matches one of BAR2 through BAR5 (even if the SG2010 drives that transaction onto the PCI bus). In other words, the SG2010 only accepts PCI transactions initiated by other PCI devices for translation into path-routed/multicast frames; it does not route incoming frames back into the fabric.

3.1.2.1.2 Segment Table Lookup

The SG2010 implements a Segment Table (described in Section 4.6.13) that assigns frame properties to incoming PCI transactions addressing a segment in the table. The SG2010 Segment Table has 1024 entries and allocates a portion of those segments to each address range defined by BAR2–BAR5. The actual number of segments per address range depends upon whether it is a 32-bit or a 64-bit range, and whether redundant routes are enabled for that range. When redundant routes are used, each segment corresponding to that address range uses two table entries; otherwise only one entry per segment is used.

The number of segments for a 32-bit address range without redundant routes is 256. A 64-bit address range has twice as many of segments as a 32-bit address range. An address range with redundant routes has half the number of segments as an address range of the same size without redundant routes. Table 3–4 lists the number of segments per address range for the allowable BAR configurations, with and without redundant routes. Redundant route entries are organized such that all primary path entries for a BAR are in the first contiguous chunk of Segment Table entries, followed by all the secondary path entries for the BAR. If a redundantly-routed BAR has 256 segments, then the first set of 256 entries contains the primary path and the second set of 256 entries contains the secondary path.

Table 3–4 Segment Allocation per BAR

	Non-redundant Routes				**Redundant Routes**			
Configuration	**BAR2**	**BAR3**	**BAR4**	**BAR5**	**BAR2**	**BAR3**	**BAR4**	**BAR5**
One 32-bit range	256	0	0	0	128	0	0	0
Two 32-bit ranges	256	256	0	0	128	128	0	0
Three 32-bit ranges	256	256	256	0	128	128	128	0
Four 32-bit ranges	256	256	256	256	128	128	128	128
One 64-bit range	512		0	0	256		0	0
Two 64-bit ranges	512		512		256		256	
One 64-bit range and one 32-bit range	512		256	0	256		128	0
One 64-bit range and two 32-bit ranges	512		256	256	256		128	128

Each segment in the Segment Table corresponds to an address slice in one of the address ranges. For example, a 32-bit address range with non-redundant routes has 256 segments. Each segment corresponds to 1/256 of the address range. If the address range is 8MB, each segment corresponds to 32KB.

The minimum supported segment size is 4KB. Therefore, the minimum supported size for a BAR with 128 segments is 512KB. The minimum supported size for a BAR with 256 segments is 1MB, and for 512 segments is 2MB.

The maximum amount of space that a 32-bit BAR can request is 2GB. The maximum amount of space that a 64-bit BAR can request is 2PB (2 Petabytes = 2^{50} bytes). This is limited by the maximum offset size of a path-routed frame.

When the SG2010 responds to a PCI transaction that matches one of its BARs, it uses a portion of the address to create an index into the Segment Table. The address bits used for this index depend on the size of the address range and the number of segments used per address range. A seven-bit index is needed to access one of 128 segments. Similarly, an eight-bit index is needed to access one of 256 segments, and a nine-bit index is needed to index one of 512 segments. For example, an eight-bit index is needed for a 32-bit address range with non-redundant routes.

The SG2010 derives the *N*-bit Segment Table index from the upper *N* bits of the address after the base address is removed. In the example of an 8MB address range, bits [32:23] comprise the base address and the next eight address bits [22:15] are the Segment Table index.

The SG2010 uses the Segment Table index to read the entry corresponding to the PCI transaction address. This entry contains the following information, needed to translate the transaction into a path-routed frame:

- Path index/Multicast ID:
  - If CoS ≠ Multicast, an index into the Path Table
  - If CoS = Multicast, this is the Multicast ID
- Output port: which fabric port the SG2010 uses when sending the frame
  - If CoS = Multicast, the Multicast Table is used to select the output port(s)
- Path length/Initial TC:
  - If CoS ≠ Multicast or Special, path length is the number of valid turns in the path. Values of 8 through 15 indicate an invalid entry.
  - If CoS = Special and Destination Channel ID = 255, indicates a Channel 255 Provisioning frame to a switch, this field is the initial Turn Count, used as:
    - > The initial Turn Count value for the frame
    - > A shift index for the path (low turns shift to higher turns)
- Class-of-service:
  - Allowed CoS values are Multicast, Asynchronous, Isochronous, High-priority Asynchronous, High-priority Isochronous, Provisioning, and Special (only when Channel ID = 255)
  - CoS = Address-routed or Special (when Channel ID ≠ 255) are illegal and result in unpredictable behavior
- Last Turn, Turn 0: The first and last turns in the path.
  - The position of the last turn is indicated by the path length – 1.
  - If there is only one turn in the path, Turn 0 is used.
- Source Channel Enable: Enables the use of source channel properties
  - Source address translation
  - Prescriptive reads
  - Write with acknowledge
- Source Channel ID
- Destination Channel ID

- MSB address: 7, 8 or 9 address offset bits, substituted in the address offset for the bits used for the segment table index.

Although the Segment Table contains the first and last turns of the path, the remainder of the path is obtained from the Path Table (described in Section 4.6.12).

3.1.2.1.2.1 Redundant Routes

If the Segment Table is configured for redundant routes for an address range, there are two Segment Table entries per segment for that address range. The first entry is used unless the entry has been marked invalid or the output port is down. An invalid entry is indicated when the path length is set to a value between 8 and 15 (inclusive); that is, if bit [3] of the path length is a 1, the entry is invalid.

If the first entry is invalid and redundant routes are enabled for that address range, then the second entry is used. If the output port for the first entry is down, but the output port for the second entry is up, then the second entry is used.

3.1.2.1.2.2 Invalid Entries

If there is not an available valid entry for a segment, then the frame is discarded and an Invalid Segment Table Entry event is signaled. This occurs either when an entry is invalid and redundant routes are not used, or when both entries in a redundant setup are invalid. The response on the PCI bus to the incoming transaction is determined by the the Master Abort Mode bit in the Bridge Control configuration register, and the PCI Target Response Mode bit in the Gateway Chip Control configuration register. If either bit is set to a 1, a target abort is returned to the PCI initiator, otherwise if both bits are 0, TRDY_L (and FFFFFFFFh for read data) is returned.

3.1.2.1.3 Path Table Lookup

The Path Table is 128 entries. Each entry contains the second (Turn1) through the sixth (Turn5) turn of the path. The Path Table index, which is one of the components of the Segment Table entry, selects the Path Table entry to be used.

After the Segment Table and Path Table lookup, the following information is associated with the PCI transaction:

- A complete path (or Multicast ID)
- A class-of-service
- A destination channel ID
- An address offset after the MSB address translation
- An operation (read request or write)

This information is used to build the frame header. The path length field in the Segment Table specifies the number of valid turns for path-routed frames. Hardware forces any unused turns to 0. Additional operations may be specified when a source channel is used.

3.1.2.1.4 Generating Channel 255 Frames

When the SG2010 generates a Channel 255 frame to a switch, the Segment Table entry can be configured to specify an initial turn count rather than a path length. This allows the SG2010 to generate Channel 255 frames with a non-zero initial turn count. This may be useful when targeting Channel 255 frames to a switch, where the turn count must equal 7 at the switch, and allows the SG2010 to generate the shortest path to a switch. Because the protocol does not support the generation of special frames, the combination of special encoding in the Segment Table entry CoS field and 255 in the Destination Channel ID field indicates that the frame is a Provisioning Channel 255 frame to a switch.

The SG2010 sets the initial turn count of the Channel 255 path-routed frame to the number specified in the Initial Turn Count field for that Segment Table entry. When reading the path from the Segment and Path Tables, the SG2010 assumes a path length of 7, which results in a path of 7 turns. The SG2010 then shifts the turns in the path by the number of turns specified by the Initial Turn Count. The SG2010 shifts in a 7 (all 1's) to the low turn position. Figure 3–5 illustrates shifting of the path based on initial turn count.

Figure 3–5 Initial Turn Count Example

Path Specification	Turn 6	Turn 5	Turn 4	Turn 3	Turn 2	Turn 1	Turn 0	Initial Turn Count
Path from Segment Table and Path Table	g	f	e	d	c	b	a	4
Path Inserted in Frame	c	b	a	7	7	7	7	4

Hardware does not force any high turns to 0 for Channel 255 paths that are shifted in this way, because the path length is assumed to be 7. Hardware forces to 7 only the low turns that are shifted in. Note that the last turn of the resulting path comes from the Segment Table (unless the initial turn count is 0), since the last turn of the unshifted path is dropped with a non-zero initial turn count.

3.1.2.1.5 Source Channel Operations

The SG2010 source channels enable the following operations:

- Address range check
- Address translation
- Ability to generate a prescriptive read and specify the amount of prescriptive read data
- Ability to generate a write with acknowledge operation

The use of a source channel is optional, and is enabled when the Source Channel Enable bit is set in the Segment Table entry. The Source Channel ID obtained from the Segment Table entry selects one of eight source channels.

The source channel's address range check ensures that the incoming address does not exceed a programmable limit. The source channel address range bits [43:12] are stored in each Source Channel Table entry (described in Section 4.6.10). Bits [43:12] of the incoming address, stripped of the base address bits but including the MSB replacement bits, are compared to this 32-bit field. If bits [43:12] of the address are greater than the address comparison field then a Source Channel Range event is signaled. If the PCI transaction was a read, a target abort is returned to the PCI initiator. If the PCI transaction was a write, the SG2010 asserts TRDY# and the write data is discarded. A write acknowledge through Event Message Unit 0 is not generated in this case if the source channel specifies a write with acknowledge operation.

If the range check is successful, the SG2010 performs a source address translation. Source address translation is the addition of a source channel address to the transaction address bits [43:12] after the MSB address bit replacement. The source channel address translation bits [43:12] are stored in each Source Channel Table entry. There is no error on an overflow – the SG2010 uses the results of the addition for the address offset regardless of whether the operation overflows. The result of this translation is the offset used in the frame header.

If the PCI transaction is a read and the Prescriptive Read Enable bit is set in the Source Channel Table entry, then the SG2010 uses a prescriptive read operation for the resulting frame. The prescriptive read amount is also specified in the Source Channel Table entry. For more information about prescriptive reads, see Section 3.3.2.4.

If the PCI transaction is a write and the Write Acknowledge Enable bit is set, the SG2010 uses a write with acknowledge operation for the resulting frame. The SG2010 handling of acknowledges in response to path-routed write or multicast frames is described in Section 3.7.3.3.1.

3.1.2.2 Fabric to PCI Address Translation

When a path-routed or multicast frame is received by the SG2010 on one of its links, the SG2010 performs a frame-to-PCI translation (unless the frame targets the SG2010 registers). The SG2010 destination channels perform path protection, offset checks, and address translation in order to derive a PCI address from a fabric offset. All incoming path-routed and multicast frames specify a destination Channel ID.

The SG2010 supports eight channels, Channel 0 through Channel 7. The Channel ID in the header of the incoming frame selects one of eight entries in the Destination Channel Table (described in Section 4.6.11). If an unsupported Channel ID is detected, the SG2010 signals an Invalid Destination Channel ID event and, if a response frame is required, returns a failure type of Channel Inactive. The incoming frame is dropped.

The destination channel can be enabled to perform a path protection check on the incoming frame. If the Protection Enable bit in the Destination Channel Table entry is set, the SG2010 compares the path and the input port of the incoming frame against two programmable path protection specifications. These path protection fields are contained in the Destination Channel Table entry. A bit associated with each path specification selects whether the path protection is to be performed against a path-routed frame or a multicast frame (using the Multicast ID of the frame). It is possible to assign one path protection field to check path-routed frames and a second path protection field to check multicast frames. If neither comparison is successful, the frame is discarded and a Destination Channel Path Protection Error event is signaled. If a response frame is returned, the Channel Lock failure type is indicated. The incoming frame is dropped.

If a destination channel is enabled to perform path protection and both path protection channels are configured to be path-routed checks, then any multicast write to that channel results in a path protection error. If the destination channel is enabled to perform path protection checks and both path protection channels are configured to be multicast ID checks, then any path-routed read or write to that channel results in a path protection error.

If the path protection check passes, a destination channel offset range check is performed. The destination channel's range check ensures that the offset of the incoming frame does not exceed a programmable limit. Bits [43:12] of the frame offset are compared to the 32-bit offset compare field. If bits [43:12] of the frame offset are greater than the offset compare field then a Destination Channel Offset Range Error event is signaled and, if a response frame is required, the Range failure type is returned. The incoming frame is dropped.

If both the path protection and offset range checks are successful, an address translation is performed on the address offset. Each destination channel entry has 52 bits of translation address corresponding to address bits [63:12]. The SG2010 adds the translation address for the selected destination channel to the frame's offset. The frame offset consists of 42 bits ([43:2]). The frame offset, when added to the destination translation address, creates a 64-bit PCI address. The translation may result in an address where the upper 32 bits are 0; in that case the SG2010 uses a 32-bit PCI address (SAC).

The SG2010 uses the translated address for the PCI transaction address. Note that the SG2010, as a PCI target, does not respond to translated PCI addresses that hit any of its address ranges, including those mapping SG2010 registers. These transactions will master abort on the PCI bus.

For more information about protocol translation, see Section 3.4.

3.1.2.3 Fabric CSR Addressing

In the StarFabric addressing model, the SG2010 registers are accessed through incoming path-routed frames using Channel 255. Whenever the SG2010 receives a path-routed Channel 255 frame, it reads or writes the register at the Channel 255 offset specified by the frame offset field. The StarFabric addressing model supports 32 bits of register address space at each node, although the SG2010 registers use only the first 32K of that space.

Channel 255 frames are also subject to path protection and offset range checks.

Registers cannot be accessed using a multicast write. Multicast writes specifying Channel 255 will result in a Channel Lock failure type if a response frame is required, and a Destination Channel Path Protection event is signaled. For more information about register mapping, register access, and register access checks, see Chapter 4.

3.1.3 Multicast Addressing

Multicast frames can be sent to multiple destinations. Multicast frames are restricted to writes; there is no multicast read. The SG2010 can be the origin or the terminus of a multicast frame. The following sections describe how multicast addresses are generated from PCI addresses as an origin and translated into PCI addresses as a terminus, and how multicast acknowledges and multicast groups are managed.

3.1.3.1 Sending Multicast Frames

Multicast frames are routed from node to node based on a table lookup at each node. The Multicast Table identifies which ports belong to a given multicast group. Multicast frames are sent out all ports that are a part of that multicast group. The multicast group of a frame is identified by the Multicast ID frame header field. See the *StarFabric Architecture Specification* for details on multicast routing.

When the address of a PCI transaction matches the base address in one of the Gateway base address registers BAR2–BAR5, a Segment Table look-up is performed as described in Section 3.1.2.1.2. If the Segment Table entry returns a multicast CoS value, then the SG2010 constructs a multicast frame. The Segment Table entry specifies the Multicast ID for the frame, which is used in place of the path specification in the frame header. The SG2010 supports Multicast IDs of 0 through 31. As with path-routed frames, the Segment Table specifies a destination channel and an optional source channel. However, the Path Table and any Segment Table fields related to path length and turn specifications are not used.

Using a source channel, an address range check and offset translation may be performed as described in Section 3.1.2.1.5. Additionally, a multicast write with acknowledge can be generated using a source channel.

The output port(s) of a multicast frame is obtained from a 32-entry Multicast Table, instead of the Segment Table as used for path-routed frames. Each Multicast Table entry corresponds to one of the 32 multicast groups supported by the SG2010 and is indexed by the Multicast ID obtained from the Segment Table entry. Each entry identifies the port(s) that are a member of the multicast group, and bits to track returning multicast acknowledge frames. The frame is sent out each port that is identified to be a member of that multicast group. It is possible to send a copy of the frame out both ports.

If no output port bits are set for that group, then that multicast group has no members. The response on the PCI bus to the incoming transaction is determined by the Master Abort Mode bit in the Bridge Control configuration register, and the PCI Target Response Mode bit in the Gateway Chip Control configuration register. If either bit is set to a 1, a target abort is returned to the PCI initiator, otherwise if both bits are 0, TRDY_L (and FFFFFFFFh for read data) is returned. The SG2010 also signals the Multicast Distribution Failure event. If the output port for a multicast group is down when the SG2010 attempts to send the multicast frame, a Multicast Distribution Failure event is set.

3.1.3.2 Receiving Multicast Frames

The SG2010 receives and translates incoming multicast frames in the same way as a path-routed frame – destination channel path protection checks (if enabled and selected for multicast), offset compare checks and offset translations are performed before the resulting transaction is initiated on the PCI bus. These destination channel operations are described in Section 3.1.2.2. The SG2010 supports all possible multicast group numbers as the terminus of a multicast frames since no lookups based on Multicast ID are performed.

If a write with acknowledge operation is specified in the header of the multicast frame received by the SG2010, then the SG2010 generates a write acknowledge frame and sends it back to the originator. The multicast write acknowledge routing is based on Multicast ID and not based on a path, so a path transform is not performed (there is no path). The CoS of the write acknowledge frame is provisioning, and the Request CoS is multicast.

3.1.3.3 Receiving Multicast Acknowledges

Because a multicast frame can have multiple destinations, a multicast write with acknowledge can generate multiple acknowledge frames. Each node must aggregate the multicast acknowledge frames for a given group before forwarding an acknowledge to the next node. However, if the multicast acknowledge frame has a Failure Type other than Normal, the acknowledge is always forwarded. For more information on aggregation of multicast acknowledges, see the *StarFabric Architecture Specification*. The SG2010 does not support tracking of multicast acknowledges with non-Normal Failure Types (multicast nack tracking).

To track whether a multicast acknowledge frame has been received on an input port, the SG2010 sets bits in the Multicast Table that correspond to each multicast group member. The SG2010 sets a multicast acknowledge bit for a port when the multicast acknowledge frame is received with its Final Multicast Ack bit set. This bit is needed to distinguish between the last acknowledge received on a port and an intermediate acknowledge with a non-Normal Failure Type.

When the last acknowledge is received, that is, a write acknowledge with the Final Multicast Ack bit has been received on each port that is a member of that multicast group, the SG2010 forwards that write acknowledge (with the Final Multicast Ack bit set) to the appropriate Event Message Unit to be written to memory, as described in Section 3.7.3.3.1. The SG2010 then clears all the acknowledge bits for that multicast group.

The SG2010 forwards all multicast acknowledge frames with error to the appropriate Event Message Unit. When the SG2010 receives a multicast acknowledge with error, if the Final Multicast Ack header bit is set the SG2010 sets the corresponding acknowledge bit in the Multicast Table as described above. If there is still an outstanding acknowledge for that multicast group, the SG2010 clears the Final Multicast Ack bit before forwarding it on. If it is the last acknowledge the SG2010 expects to receive, the SG2010 sets the Final Multicast Ack bit.

When the SG2010 sends a multicast write with acknowledge into the fabric, it clears all Multicast Ack bits in the Multicast Table entry corresponding to that group.The SG2010 also implements a Clear Multicast Ack control bit for each Multicast Table entry. When software writes 1 to this bit, all Multicast Acknowledge bits for that group are cleared.

If the SG2010 receives a multicast write acknowledge frame with an unsupported Multicast ID and a Failure Type of Normal, it discards the frame and takes no other action. If the multicast write ack contained a non-Normal failure type, it directs the write acknowledge to the specified EMU.

3.1.3.4 Configuring Multicast Groups and Bandwidth Allocation

Unlike switches, which set up and tear down multicast groups in hardware based on provisioning frames, the SG2010 multicast groups in the Multicast Table are managed by software.

The SG2010 implements the Bandwidth Allocation Counter in the Link State Table (described in Section 4.6.7) to track requested bandwidth on a link. Again, this feature is managed in hardware by switches. However, the SG2010 bandwidth counter is managed by software and not by SG2010’s hardware. It has no operations associated with it, a functions essentially like a scratchpad register. The SG2010 does not perform any bandwidth operations when it sends bandwidth reservation frames or receives bandwidth responses.

However, the Software Generate Frame (SGF) function of the SG2010 is used to initiate bandwidth reservation frames in order to set up multicast groups and reserve bandwidth within the fabric. When the SG2010 is the origin of a bandwidth reservation frame, the SGF function waits for a bandwidth response before signaling that it is done. The bandwidth response is handled like any other SGF response and written to PCI local memory. The Secondary Operation field containing the completion status is stored in the SGF Control register. For a description of the SGF function, see Section 3.11.

When the SG2010 is the terminus of a provisioning frame that requests bandwidth allocation or a multicast join-group operation, it always returns a positive response to the origin. Reception of this frame by the SG2010 indicates that it was successfully propagated along the path. When the SG2010 is the terminus of a provisioning frame that frees bandwidth, exits a multicast group, or tears down a multicast group, it simply discards the frame because no operation is required, nor is a response frame returned.

3.1.4 Address Decoding in the PCI Addressing Model

This section describes how PCI transaction addresses are decoded for generation of address routed frames, and decoding of incoming address routed frames from the fabric. The Bridge function performs decoding, translation, and forwarding for address-routed frames.

Bridge address decoding of incoming PCI transactions is done for two purposes. One is to determine whether to accept the transaction on the PCI bus; that is, assert DEVSEL_L. The other is to select the output port for the resulting frame.

For frames incoming from the fabric, address decoding selects whether the target is SG2010 registers or the PCI bus.

The Bridge function implements the standard set of PCI-to-PCI bridge configuration registers, which include base and limit registers, bus numbers, and control bits used for address decoding. These registers are used to determine whether to accept a transaction on the PCI bus.

To select the output port for a frame, the SG2010 implements shadow copies of each of its link partners' address-decode configuration registers. These copies are stored in the Port Map Table (described in Section 4.6.2). The Port Map Table decode can be thought of as the PCI-to-PCI bridge address decode at the next level of hierarchy. Each port has a corresponding Port Map Table entry.

During fabric enumeration, the SG2010 enables the Port Map Tables that are associated with ports in the PCI hierarchy by setting the Port Map Table Enable bit in the Port State Table (described in Section 4.6.6). Software can also enable Port Map Tables by using a register write to set the Port Map Table Enable bit. For more information about fabric enumeration, see Section 3.8.4.

3.1.4.1 Smart Address Routing

Smart address routing allows shorter address-routed paths for memory and I/O traffic. It allows routing along a link that is not a part of the PCI hierarchy, bypassing the PCI hierarchical path.

Software can enable a port for smart address routing by setting the Smart Address Enable bit. Both the Port Map Table Enable bit and the Smart Address Enable bit must be set to allow smart address routing. Setting the Smart Address Enable bit disables the decoding of configuration transactions against the Port Map Table, while still allowing memory and I/O transactions to be decoded. Software must also configure the Port Map Table registers.

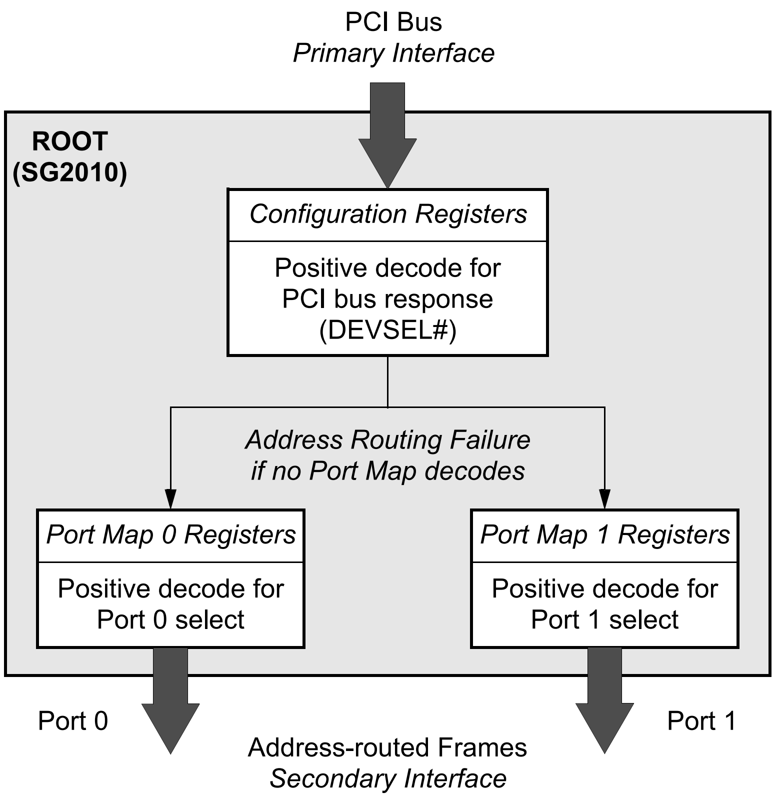
When the SG2010 is a root, it is possible to set up smart address routing where two Port Map Table ranges overlap. In this case, if an address is successfully decoded against both Port Map Tables, the SG2010 forwards the transaction out of the port with the Smart Address Enable bit set.

Note: *Other cases of overlapping Port Map Table ranges are not supported and unpredictable results can occur.*

3.1.4.2 Downstream Address Decoding as a Root

When the SG2010 is a root, PCI transactions are accepted from the PCI bus based on the positive address decode against the Bridge's configuration register ranges and control bits. If this positive decode is not successful, the SG2010 does not assert DEVSEL_L. If the decode is successful and the transaction is accepted, then the SG2010 positively decodes the address against the Port Map Table register ranges to select the output port. To perform the decode against a Port Map Table Entry, the corresponding Port Map Table Enable bit must be set. If none of the Port Map Table decodes are successful, then an Address Routing Failure event is signaled (if not a configuration transaction) and the SG2010 sets the Received Master Abort bit in the Secondary Status configuration register. Figure 3–6 shows how the SG2010 as a root performs downstream address decoding.

Figure 3–6 Downstream PCI Address Decoding as a Root Bridge


PCI Bus
Primary Interface
ROOT (SG2010)
Configuration Registers
Positive decode for PCI bus response (DEVSEL#)
Address Routing Failure if no Port Map decodes
Port Map 0 Registers
Positive decode for Port 0 select
Port Map 1 Registers
Positive decode for Port 1 select
Port 0
Port 1
Address-routed Frames
Secondary Interface

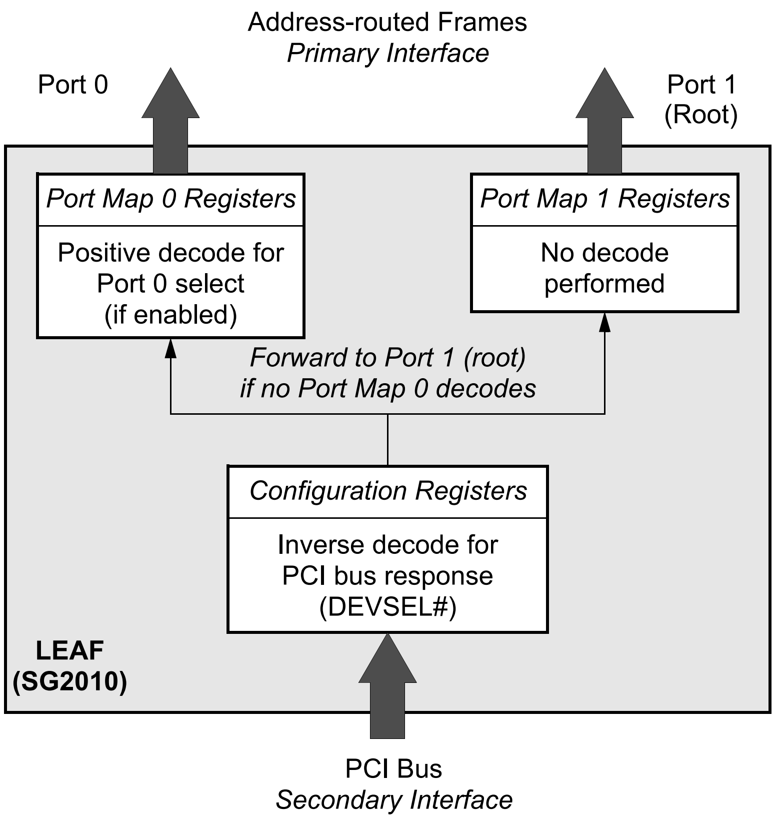
3.1.4.3 Upstream Address Decoding as a Root

When the SG2010 is a root, upstream address-routed frames are received on the fabric interface and directed to the PCI interface. The SG2010 inversely decodes the offset of the address-routed frame against the appropriate address ranges defined in the Bridge configuration registers. If the inverse decode is unsuccessful, the frame is discarded, an Address Routing Failure event is signaled (if not a configuration frame), and a failure type of Software Routing Failure is returned in a response frame (if required). If the inverse decode is successful, a positive decode is performed against the Gateway's BAR0 and BAR1 to determine if a register access is to be performed. If the frame's address offset does not hit the Gateway's BAR0 or BAR1, then the frame is translated into a PCI transaction and driven on the PCI bus.

3.1.4.4 Upstream Address Decoding as a Leaf

When the SG2010 is a leaf, the SG2010 inversely decodes upstream PCI transactions against the Bridge's register ranges and control bits in configuration space. If this inverse decode is not successful, the SG2010 does not assert DEVSEL_L on the PCI bus. If the inverse decode is successful, the SG2010 positively decodes the address against all enabled Port Map Table ranges. If a Port Map Table decode is successful, the resulting frame is sent out that port. If all Port Map Table decodes are unsuccessful, the frame is sent out the root port to the next node. An Address Routing Failure event cannot occur, since the frame is sent by default to the root port. Figure 3–7 shows how the SG2010 as a leaf performs upstream address decoding

Figure 3–7 Upstream Address Decoding as a Leaf Bridge



3.1.4.5 Downstream Address Decoding as a Leaf

When the SG2010 is a leaf, downstream address-routed frames are received on the fabric interface and directed to the PCI interface. Because the downstream transaction was positively decoded against SG2010’s parent node’s Port Map Table (which is a copy of SG2010’s register ranges in configuration space), an address decode failure cannot occur. Instead, a positive decode is performed against the Gateway’s BAR0 and BAR1 to determine whether the frame is a register access. If the frame’s offset does not hit the Gateway’s BAR0 or BAR1, then the frame is translated into a PCI transaction on the PCI bus.

3.1.4.6 PCI Configuration Address Decoding

This section describes the decoding and routing of PCI configuration transactions. Configuration transactions are forwarded through the fabric as address-routed frames based on bus number decodes.

Configuration address decoding uses the following registers, in either the SG2010 configuration space or Port Map Tables:

- Primary Bus Number
- Secondary Bus Number
- Subordinate Bus Number

When the Secondary and Subordinate Bus Number registers are 0, it is assumed that they have not been configured and any positive decode against them is considered unsuccessful, whether it is against the configuration registers or the Port Map Table. Additionally, Port Map Table decodes are considered unsuccessful if the Smart Address Enable bit is set.

3.1.4.6.1 Configuration Register Accesses as a Root

When the SG2010 PCI interface is the primary bus interface, the Bridge configuration space and the Gateway configuration space are accessed through a Type0 configuration transaction from the PCI bus.

If the Bridge is enabled, Bridge configuration space is accessed when a Type0 configuration transaction specifying Function0 is detected and the IDSEL signal is asserted high. Gateway configuration space is accessed when a Type0 configuration transaction specifying Function1 is detected and the IDSEL signal is asserted high. If the Bridge is enabled, the SG2010 does not respond to configuration transactions specifying functions other than 0 or 1, even if the IDSEL signal is asserted.

If the Bridge is not enabled, then the Gateway configuration registers are accessed when a Type0 configuration transaction is detected and the IDSEL signal is asserted. In this case the function number is not decoded.

When a root, the SG2010 does not support configuration register accesses through a configuration address-routed frame from the fabric interface.

3.1.4.6.2 Configuration Register Accesses as a Leaf

If the Bridge is enabled and the SG2010 receives a Type0 configuration frame on its fabric interface, a register access to Bridge configuration space is performed. The only additional address decoding required is using the Register Number to select the register. Because the Bridge is a single function device in this mode, the function number decode is not performed. Any Function Number results in a Bridge configuration register access. The Bridge's configuration space cannot be accessed from the secondary bus.

Register access to Gateway configuration space is performed using a Type0 configuration access on the secondary bus. This access can be from either the PCI interface or, if the Bridge is enabled, the fabric interface. From the PCI interface, a Type0 PCI configuration transaction is initiated on the PCI bus with the SG2010 IDSEL pin asserted. Gateway configuration space can be accessed from the fabric interface when a Type1 configuration frame is received with a bus number equal to the secondary bus number and the device number is equal to the Gateway's device number. The Bridge function performs this address decoding. In both cases the function number is not decoded.

3.1.4.6.3 Downstream Configuration Decoding as a Root

When the SG2010 is a root (PCI bus is primary), downstream configuration transactions enter the SG2010 from the PCI bus and exit through the fabric interface as address-routed frames.

The SG2010 responds (asserts DEVSEL_L) to Type1 configuration transactions for downstream forwarding when the bus number falls between its secondary and subordinate bus numbers (inclusive). If the bus number is equal to the secondary bus number, the SG2010 translates the transaction into a Type0 configuration frame on the secondary bus (the fabric interface). If the device number is 0, the transaction is converted into a PCI frame specifying Type0 configuration and sent to Port0. If the device number is 1, the Type0 configuration PCI frame is sent to Port1. If any other device number is specified, or the corresponding port is not up, an Address Routing Failure event is signaled.

If the bus number is equal to the secondary bus number and the PCI special cycle encoding is detected (Register Number = 00h, Function Number and Device Number are all 1s), then a Fabric Special Cycle event occurs. The frame is discarded and if a response frame is required, a Normal failure type is used.

If the bus number is greater than the secondary bus number and less than or equal to the subordinate bus number, then the transaction is forwarded as an address-routed frame specifying a Type1 configuration. In order to select the output port for the frame, the SG2010 performs a positive decode against all Port Map Table Secondary and Subordinate Bus Number registers. If none of the positive decodes against the Port Map Table ranges are successful, then an Address Routing Failure event is signaled.

When an Address Routing Failure occurs for these cases, the SG2010 discards write data or responds with FFFFFFFFh to the initiator and sets the Master Abort Detected Status bit in the Secondary Status register. An Address Routing Failure event is not signaled for configuration transactions.

3.1.4.6.4 Upstream Configuration Decoding as a Root

When the SG2010 is a root, an upstream configuration transaction enters the SG2010 through the fabric interface as an address-routed frame and exits as a PCI transaction.

The SG2010 forwards upstream only those Type1 configuration write transactions that are to be converted to a PCI special cycle (device number and function number are 1s and register number is 0s). If this special cycle encoding is not detected on the upstream transaction, an Address Routing Failure is signaled. If the special cycle encodings are detected, the SG2010 inversely decodes the bus number of the incoming frame against the range defined by the Secondary and Subordinate Bus Number registers in configuration space. If the bus number is not successfully inversely decoded, then an Address Routing Failure event is signaled.

When an Address Routing Failure event is signaled in these cases, the SG2010 discards write data or responds with FFFFFFFFh in the read completion frame. A failure type of Software Routing Failure is used if a response frame is required. An Address Routing Failure event is not signaled for configuration transactions.

If the bus number is inversely decoded successfully and the special cycle encoding is satisfied, the SG2010 compares the bus number against its primary bus number. If the bus numbers match, the SG2010 initiates the transaction on the PCI bus as a Special Cycle. Otherwise, the SG2010 initiates the transaction on the PCI bus as a Type1 configuration transaction.

3.1.4.6.5 Downstream Configuration Decoding as a Leaf

When the SG2010 is a leaf (PCI is secondary), a downstream configuration transaction enters the SG2010 as a configuration address-routed frame on the fabric interface and exits as a PCI transaction on the PCI bus.

When the SG2010 receives a Type1 configuration frame on its fabric interface, its parent node has performed the positive bus number decoding (through its Port Map Table decode).

The SG2010 compares the bus number in the frame offset against its Secondary Bus Number register in configuration space to determine whether to convert the transaction to a Type0 configuration transaction on the PCI bus. If the bus number matches the secondary bus number, then the transaction is intended either for the Gateway or for a device on the SG2010 secondary bus. If the configuration frame is encoded to generate a special cycle, then the SG2010 initiates a PCI Special Cycle on the PCI bus.

If a special cycle encoding is not detected, the SG2010 compares the device number to the Gateway's device number. The Gateway's device number is programmable using the Chip Control configuration register (can be initialized through serial preload). If the device numbers match, a Gateway configuration access is performed. If the device number is not equal to the Gateway's device number, then the SG2010 initiates the transaction on the PCI bus as a Type0 configuration transaction.

If the bus number does not match the secondary bus number then the SG2010 initiates the transaction as a Type1 configuration transaction on the PCI bus.

3.1.4.6.5.1 Hiding Downstream Resources from the Host Processor

When a leaf, the SG2010 implements three mechanisms to hide local resources from the host processor:

- IDSEL mask to selectively hide secondary bus devices
- Gateway mask bit to hide the SG2010 Gateway function
- BAR2–BAR5 mask bit to hide memory resources requested by these gateway BARs

The SG2010 implements an IDSEL mask that selectively blocks the assertion of IDSEL to secondary bus PCI devices. The AD[31:16] signal pins are used as IDSEL signals during the address phase of Type0 configuration transactions on the PCI bus. During a Type0 transaction, only one of these signals is asserted to select the target of the transaction. The SG2010 gates the AD[31:16] lines with the 16-bit IDSEL mask when it initiates a Type0 configuration transaction. If the mask bit that corresponds to an asserted AD line is set, then the signal is not asserted on the PCI bus. The resulting configuration transaction with a blocked IDSEL master aborts on the PCI bus. If the IDSEL signal pin of a secondary bus device is connected to the blocked AD line, that secondary bus device is then hidden from the host processor.

The Gateway function is hidden when the Gateway Mask bit is set in the Chip Control configuration register (described in Section 4.8.8.1); that is, Gateway configuration registers are not accessible through downstream configuration frames. The SG2010 does not initiate a transaction on the PCI bus, but returns a response frame to the origin with a failure type of Software Routing Failure. In both cases the Master Abort Received bit is set in the Secondary Status register. Gateway configuration registers are still accessible through their dual-mapped locations (memory, I/O or Channel 255 space) or from the PCI bus.

Additionally, the Gateway’s BAR2, 3, 4 and 5 registers can be hidden from the host if the HIDEBARS bit is set in the Gateway’s Chip Control configuration register. In this case, when the Gateway BAR2 through BAR5 registers are read through a downstream configuration read, 0s are returned, indicating to the host processor that the BAR is disabled. A configuration read of these registers from the secondary bus is completed as usual to allow configuration by a local processor or through application-specific means. These BAR registers are not hidden when accessed through their dual-mapped locations (that is, through memory, I/O, or Channel 255 space).

3.1.4.6.6 Upstream Configuration Decoding as a Leaf

When the SG2010 is a leaf, an upstream configuration transaction enters the SG2010 through the PCI bus and exits through the fabric interface as an address-routed frame. Only those configuration transactions to be converted to special cycles are forwarded upstream. When the SG2010 detects a Type1 configuration write transaction on its secondary interface, it decodes the following in order to assert DEVSEL_L:

- Bus number is less than the secondary bus number or greater than the subordinate bus number (that is, inversely decoded)
- Function code is all 1s
- Device number is all 1s
- Register number is all 0s

The SG2010 then compares the bus number to its primary bus number. If the bus number matches the primary bus number, the SG2010 signals a Fabric Special Cycle event and discards the frame. A normal completion (TRDY_L) is returned to the initiator.

If the bus number of the transaction does not match the Primary Bus Number, the SG2010 performs a positive decode against the enabled Port Map Tables to select the output port. If the positive decode is not successful, then the frame is forwarded to the root port.

3.1.4.7 PCI Memory Address Decoding

This section describes decoding and routing PCI memory transactions using the Bridge function. Memory transactions are forwarded through the fabric as address-routed frames based on base and limit address decodes. Forwarding memory transactions as path-routed frames is described in Section 3.1.2

When forwarding memory transactions and frames, the SG2010 decodes memory addresses against the following registers, either in SG2010's configuration space or Port Map Table:

- Memory base and limit registers, defining a 32-bit address range in non-prefetchable space
- Prefetchable memory base and limit registers, defining a 64-bit address range in prefetchable space
- If the VGA Enable bit is set in the Bridge Control register, the 32-bit VGA memory addresses 000A0000h – 000BFFFFh

For positive address decodes (downstream), the corresponding Memory Enable bit must be set in the Command register (described in Section 4.7.1.3) in configuration space or the Port Map Table. For inverse address decodes (upstream), the Master Enable bit must be set in the Command register in configuration space; otherwise, a master abort occurs.

When decoding an address from an address-routed frame, if the target region is I/O, configuration, or SAC, a 32-bit address decode is performed. Offset bits [49:32] are not decoded. If the target region is a DAC, all offset bits are decoded.

3.1.4.7.1 Gateway Register Accesses in Memory Space

The SG2010 memory-mapped registers are associated with the Gateway function through BAR0. When the SG2010 detects a memory transaction on the PCI bus that has a base address matching BAR0's base address, the SG2010 performs a memory-mapped register access. This applies whether the SG2010 is a leaf or a root.

Additionally, these registers may also be accessed from the fabric interface when an address-routed memory frame is received that has an offset matching BAR0's base address.

3.1.4.7.2 Downstream Memory Address Decoding as a Root

When the SG2010 is a root, a downstream memory transaction enters the SG2010 from the PCI bus and exits as an address-routed frame to the fabric. The SG2010 positively decodes memory transactions against the SG2010 memory range registers in configuration space to determine whether to assert DEVSEL_L on the PCI bus.

After the SG2010 asserts DEVSEL_L, it determines which output port to use by positively decoding against the Port Map Table memory range registers and control bits.

If there are no successful Port Map Table decodes, an Address Routing Failure event is signaled. The response on the PCI bus to the incoming transaction is determined by the the Master Abort Mode bit in the Bridge Control configuration register, and the PCI Target Response Mode bit in the Gateway Chip Control configuration register. If either bit is set to a 1, a target abort is returned to the PCI initiator, otherwise if both bits are 0, TRDY_L (and FFFFFFFFh for read data) is returned. In both cases the Master Abort Received bit is set in the Secondary Status register; if a target abort is signaled the Signaled Target Abort event bit is set in the Status register in Bridge configuration space.

3.1.4.7.3 Upstream Memory Address Decoding as a Root

When the SG2010 is a root, an upstream address-routed frame enters from the fabric and exits on the PCI bus.

The SG2010 inversely decodes the frame address offset against the memory ranges defined by the SG2010 configuration registers to determine whether to forward the frame. If the inverse decode is not successful, an Address Routing Failure event is signaled. If a response frame is required, it is sent back to the origin with a failure type of Software Routing Failure. If the operation is a memory write without acknowledge then no response frame is returned, but the Master Abort on Write w/o Ack event bit is also signaled.

If the address is inversely decoded successfully, then the SG2010 compares the address offset against the Gateway's BAR0 base address to determine whether to perform a CSR access. If this compare is not successful, the SG2010 initiates a PCI transaction on the PCI bus. The SG2010 does not compare the address against the Gateway's BARs 2–5.

3.1.4.7.4 Downstream Memory Address Decoding as a Leaf

When the SG2010 is a leaf, a downstream memory address-routed frame enters through the fabric interface and exits as a PCI transaction on the PCI bus.

Because SG2010's parent node has performed the positive decode against its Port Map Table registers, the SG2010 needs to determine only whether the transaction is intended for the Gateway BAR0 or the PCI bus. If the SG2010 successfully decodes the address offset against the Gateway's BAR0 base address register, then a CSR access is performed. Otherwise, the SG2010 initiates the PCI transaction on the PCI bus.

3.1.4.7.5 Upstream Memory Address Decoding as a Leaf

When the SG2010 is a leaf, an upstream transaction enters from the PCI bus and exits as an address-routed frame to the fabric.

When the SG2010 detects a memory transaction on the PCI bus, it inversely decodes the transaction against its Bridge memory ranges in configuration space to determine whether to assert DEVSEL_L.

The SG2010 also performs an optional additional decode to condition DEVSEL_L assertion. The SG2010 implements a device-specific secondary bus address window that may be used to block the forwarding of inversely decoded transactions. This secondary bus address window is defined by the Secondary Base Address register and the Secondary Limit Address register. These registers are mapped in Gateway device-specific configuration space. Control bits in the Gateway's Chip Control configuration register determines how this address range is used. The secondary bus range may be disabled, or it may be configured to block all inversely decoded transactions that fall inside the window, or to block all inversely decoded transactions that fall outside the window.

Therefore, in order to assert DEVSEL_L in response to a PCI transaction on the secondary bus, the address must be successfully inversely decoded against the Bridge's configuration memory address ranges, and also must not be blocked by the secondary bus address window.

If the above decodes are successful, the SG2010 positively decodes the address against the enabled Port Map Table memory ranges to select an output port. If there are no successful Port Map Table address decodes, then the frame is forwarded to the root port.

3.1.4.7.6 PCI 64-bit Address Support

Address-routed frames support 50 address bits. In order to support 64-bit PCI addressing, it is assumed that the PCI system composed of the fabric and the PCI devices attached to its endpoints are mapped in the same 50-bit address region. However, this region may be located anywhere in 64-bit address space. In other words, bits [63:50] may be any value, but all nodes in the fabric must use this value for 64-bit addresses.

64-bit addresses (DACs) are decoded only against the Prefetchable Memory Base and Limit registers. The Memory Base and Limit registers are restricted to 32-bit address decodes. After then PCI transaction is successfully decoded against the Prefetchable Memory Base and Limit register, the upper 14 bits of the address are removed and only the lower 50 bits are used for the address-routed frame address field.

To resolve address decoding ambiguity caused by the removal of the upper address bits, the frame header Target Region field indicates whether the address-routed frame originated from a 32-bit address (SAC, or single-address cycle) or a 64-bit address (DAC, or dual-address cycle) PCI transaction.

A frame offset with a 64-bit address specified in its Target Region field is only decoded against the Prefetchable Memory Base and Limit address registers; all other decodes (32-bit memory and VGA) are considered unsuccessful.

A frame offset with a 32-bit address specified in its Target Region field is assumed to have all zeros in its upper address bits [63:32] for the purposes of comparing against the Prefetchable Memory Base and Limit address registers.

When a frame is translated into a PCI transaction and a DAC Target Region is specified, then the upper 14 bits of the 64-bit address must be replaced at the terminus. When a dual-address transaction is initiated on the PCI bus, the upper 14 bits in the Prefetchable Memory Base Upper 32 Bits register are used for the upper 14 address bits on the outgoing address.

3.1.4.8 PCI I/O Address Decoding

This section describes the decoding and routing of PCI I/O transactions using the Bridge function. I/O transactions are forwarded through the fabric as address-routed frames based on base and limit address decodes. I/O transactions cannot be translated to path-routed frames.

When forwarding I/O transactions and frames, the SG2010 decodes I/O addresses against the following registers, either in the SG2010 configuration space or Port Map Table:

- I/O base and limit registers, defining a 32-bit I/O address range.
  - If the ISA Mode bit is set in the Bridge Control register, only the low 256 bytes in each 1KB chunk is positively decoded (bits [9:8] must be 00b)
    - Only applies to I/O addresses in the first 64Kbytes of I/O space
- If the VGA Enable bit is set in the Bridge Control register, the VGA I/O address bits [9:0] = 3B0 – 3BBh and 3C0 – 3DFh
  - If the VGA 16-bit Decode bit is set, address bits [15:10] must be 0, otherwise they can be any value. The VGA 16-bit Decode Support bit must be set to activate this bit. The VGA 16-bit Decode Support bit is located in the Chip Control register in Gateway configuration space.
  - Address bits [63:16] must be zero
- If the VGA Snoop bit is set, the VGA I/O address bits [9:0] = 3C6h, 3C8h, and 3C9h
  - If the VGA 16-bit Decode bit is set, address bits [15:10] must be 0; otherwise they can be any value. The VGA 16-bit Decode Support bit must be set to activate this bit. The VGA 16-bit Decode Support bit is located in the Chip Control register in Gateway configuration space.
  - Address bits [63:16] must be zero

For positive address decodes (downstream), the corresponding I/O Enable bit must be set in the Command register in configuration space or the Port Map Table. For inverse address decodes (upstream), the Master Enable bit must be set in the Command register in configuration space; otherwise, a master abort occurs.

3.1.4.8.1 Gateway Register Accesses in I/O Space

The SG2010 I/O-mapped registers are associated with the Gateway function through BAR1. When the SG2010 detects an I/O transaction on the PCI bus that has a base address matching BAR1’s base address, the SG2010 performs an I/O-mapped register access. This applies whether the SG2010 is a leaf or a root.

Additionally, these registers may also be accessed from the fabric interface when an address-routed I/O frame is received that has an offset matching BAR1's base address.

3.1.4.8.2 Downstream I/O Address Decoding as a Root

When the SG2010 is a root, a downstream I/O transaction enters the SG2010 from the PCI bus and exits as an address-routed frame to the fabric. The SG2010 positively decodes I/O transactions against its memory range registers in configuration space to determine whether to assert DEVSEL_L on the PCI bus.

Once the SG2010 asserts DEVSEL_L, it determines which output port to use by positively decoding against the Port Map Table I/O range registers and control bits.

If there are no successful Port Map Table decodes, an Address Routing Failure event is signaled. The response on the PCI bus to the incoming transaction is determined by the the Master Abort Mode bit in the Bridge Control configuration register, and the PCI Target Response Mode bit in the Gateway Chip Control configuration register. If either bit is set to a 1, a target abort is returned to the PCI initiator, otherwise if both bits are 0, TRDY_L (and FFFFFFFFh for read data) is returned. In both cases the Master Abort Received bit is set in the Secondary Status register. If a target abort is returned, the Signaled Target Abort bit is set in the Status register in Bridge configuration space.

3.1.4.8.3 Upstream I/O Decoding as a Root

When the SG2010 is a root, an upstream address-routed frame enters from the fabric and exits on the PCI bus.

The SG2010 inversely decodes the frame address offset against the I/O ranges defined by SG2010's configuration registers to determine whether to forward the frame. If the inverse decode is not successful, an Address Routing Failure event is signaled. If a response frame is required, it is sent back to the origin with a failure type of Software Routing Failure. If the operation is a write without acknowledge then no response frame is returned, but the Master Abort on Write w/o Ack event bit is also signaled.

If the address is inversely decoded successfully, then the SG2010 compares the address offset against the Gateway's BAR1 base address to determine whether to perform a CSR access. If this compare is not successful, the SG2010 initiates a PCI transaction on the PCI bus.

3.1.4.8.4 Downstream I/O Decoding as a Leaf

When the SG2010 is a leaf, a downstream I/O address-routed frame enters from the fabric and exits as a PCI transaction on the PCI bus.

Since SG2010's parent node has performed the positive decode against its Port Map Table registers, the SG2010 only needs to determine whether the transaction is intended for the Gateway BAR1 or the PCI bus. If the SG2010 successfully decodes the address offset against the Gateway's BAR1 base address register, then a CSR access is performed. Otherwise, the SG2010 initiates the PCI transaction on the PCI bus.

3.1.4.8.5 Upstream I/O Decoding as a Leaf

When the SG2010 is a leaf, an upstream transaction enters from the PCI bus and exits as an address-routed frame to the fabric.

When the SG2010 detects an I/O transaction on the PCI bus, it inversely decodes the transaction against its Bridge I/O ranges in configuration space to determine whether to assert DEVSEL_L.

If the inverse decode is successful, the SG2010 positively decodes the address against the enabled Port Map Table I/O ranges to select an output port. If there are no successful Port Map Table address decodes, then the frame is forwarded to the root port.

3.1.4.9 Address Decoding Enable Summary

Table 3–5 summarizes the address decoding enables.

Table 3–5 Address Decoding Enable Summary

Register	Bit	Control
Gateway Command configuration register	Memory Enable	Enables BAR0, BAR2 through BAR5 address decode Enables Expansion ROM BAR decode (bit [0] of BAR must also be 1)
	I/O Enable	Enables BAR1 address decode
	Master Enable	Enables initiation of PCI memory and I/O transactions by Gateway
Bridge Command configuration register	Memory Enable	Enables memory base/limit and VGA positive decodes on PCI
	I/O Enable	Enables I/O base/limit and VGA positive decodes on PCI
	Master Enable	Enables all memory and I/O inverse decodes (PCI or fabric interface)
Port Map x Command register	Memory Enable	Enables Port Map memory base/limit and VGA positive decodes
	I/O Enable	Enables Port Map I/O base/limit and VGA positive decodes
	Master Enable	Not used – Port Maps only used for positive decode

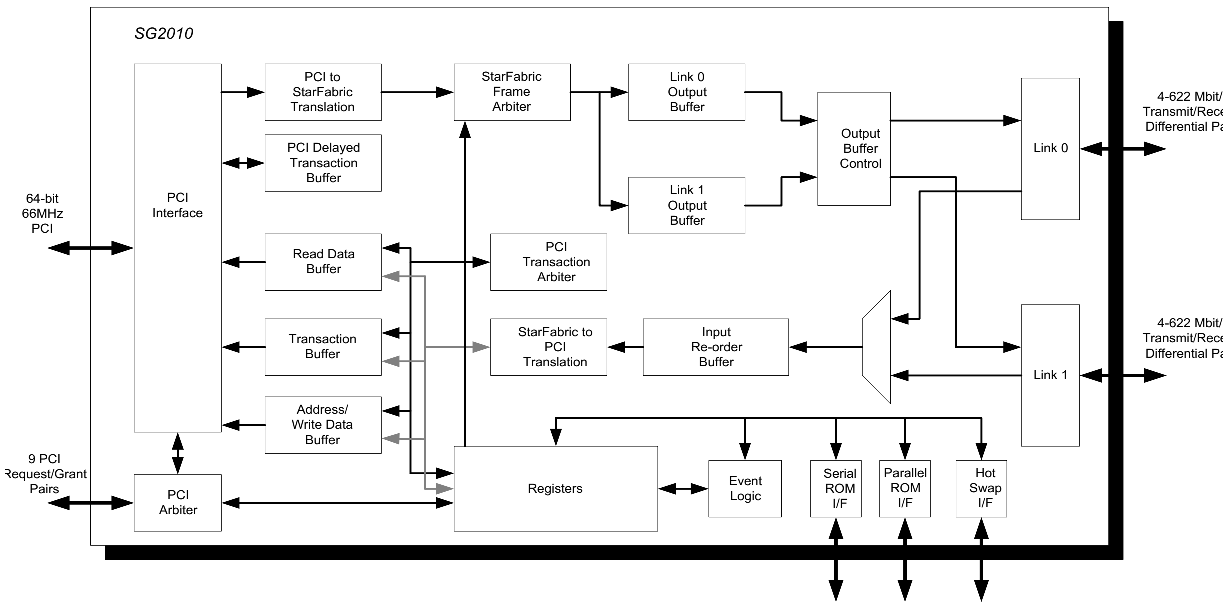
3.2 Buffer Management

This section describes the management of SG2010 data buffers. A block diagram of the SG2010 data paths is shown in Figure 3–8.

Traffic flowing from the StarFabric interface to the PCI interface is regulated through the use of the StarFabric line credit mechanism. That is, the link partner sending frames to the SG2010 must have enough line credits (representing available SG2010 buffer space) to accommodate the size of the frame. Line credits are discussed in Section 3.2.2 and more fully in the *StarFabric Architecture Specification*.

This section is predominantly concerned with the conditions for regulating traffic flow from the PCI interface to the StarFabric interface. This is done through the PCI target retry mechanism. If a PCI bus master initiates a transaction to the SG2010 and it cannot accept the transaction due to insufficient space in either its own buffers or its link partner's buffers, then the SG2010 returns a target retry to the PCI bus master

Figure 3–8 SG2010 Block Diagram



There are several components to SG2010 buffer management for traffic flowing from the PCI bus to the StarFabric interface, as shown below.

Link Output Buffer Threshold	Determines whether there is enough room in the Link Output buffer to accommodate a frame translated from a PCI transaction.
PCI Delayed Transaction Threshold	Determines whether there is enough room in the 8-entry PCI Delayed Transaction buffer to hold an incoming delayed transaction (reads, I/O writes, configuration writes).
Read Data Threshold	Determines whether there is enough room in the Read Data Buffer to accommodate the read data that would be returned to the SG2010 in response to a read. Gates the acceptance of a read request transaction from PCI.
Line Credits	Determines whether enough buffer space of the correct type exists in the link partner to accommodate a frame translated from a PCI transaction. This mechanism is specified by the StarFabric protocol, although line credit amounts are implementation specific.

3.2.1 SG2010 Buffer Thresholds

Before the SG2010 can queue a PCI transaction, the SG2010 determines whether it has enough space in its own buffers to accommodate the transaction. The SG2010 performs this check in addition to the line credit check, which determines whether its link partner has enough buffer space to accept the resulting frame.

3.2.1.1 Link Output Buffer Thresholds

Each link has a Link Output buffer that stores frames destined for the fabric. All transactions moving from PCI to a link pass through this buffer. The SG2010 checks that there is sufficient space in the Link Output buffer before queueing a PCI transaction. If there is insufficient space, the SG2010 returns target retry to the PCI transaction and does not queue it. In order to prevent starvation of larger transactions by smaller transactions, all transactions require that the same amount of buffer space be available. For more information about using line credits to prevent starvation, see Section 3.2.2.4.

3.2.1.2 PCI Delayed Transaction Buffer Thresholds

If the PCI transaction is a read or an I/O or configuration write, the SG2010 must check to see that an appropriate entry is available in the PCI Delayed Transaction buffer. This buffer holds delayed transaction information for incoming PCI transactions directed to the fabric. There are eight entries in the PCI Delayed Transaction buffer.

The Delayed Transaction Buffer Mode bit in the Gateway Chip Control configuration register specifies how the SG2010 manages these buffer entries. If a CoS reservation mechanism is enabled with this bit, three entries are dedicated for delayed transactions translated to specific classes-of-service – one entry for address-routed/asynchronous, one for isochronous/HP-Isochronous, and one for HP-asynchronous/provisioning. Five of those entries can be used for any incoming PCI delayed transaction. If the CoS reservation mechanism is disabled, all eight entries can be used for any incoming PCI delayed transaction.

3.2.1.3 Read Data Thresholds

Before the SG2010 can queue a PCI read transaction, it reserves space in the Read Data Buffer for read data to be returned to the initiator. This guarantees that the forward progress of an outstanding memory read transaction is not gated by a lack of read buffer space at the origin of the read request frame. The maximum amount of read data buffer space that can be reserved is 514 Dwords (2KB + 8 bytes). The minimum amount of space that can be reserved is two Dwords. The read data reservation operates on quadword (two Dwords) aligned address boundaries. For example, if a 2KB transfer is requested but the address starts on an odd Dword boundary, an additional eight bytes must be reserved as well.

If there is not enough read buffer space to accommodate the amount of data to be requested by the read at the terminus, the SG2010 returns target retry to the PCI transaction and does not queue the transaction. SeeSection 3.3.2 for a description of how read request amounts are determined.

3.2.2 Line Credits

The SG2010 and all StarFabric compliant devices use a credit algorithm to manage data buffer usage for frames sent between StarFabric nodes. This line credit algorithm guarantees that no frame is sent unless there is buffer space available in its link partner to accommodate it. Details of the line credit mechanism are described in the *StarFabric Architecture Specification.*

3.2.2.1 Line Credit Types

Line credit types are divided into class-of-service credits and turn credits. Although the protocol allows seven possible classes–of-service, the SG2010 supports four CoS credit types. The SG2010 aliases the remaining three classes-of-service for line credit and ordering purposes. The SG2010 four CoS credit types are:

- Address-routed/asynchronous
- Isochronous/HP-Isochronous
- Multicast
- Provisioning/HP-asynchronous

The first CoS in each pair is the CoS used in credit debiting, returning credits, and receiving credits.

The SG2010 supports eight sets of turn credits for each of the eight possible turn values in the next node. A turn is a port number relative to the input port on that component. For more information, see the *StarFabric Architecture Specification*. Because the SG2010 does not route frames back into the fabric, there is only one type of turn credit used for frames transmitted to the SG2010 – turn 0 credits, which direct the frame to PCI bus. The only type of turn credits the SG2010 returns are turn 0 credits.

The CoS line credit types must only be used for frames with the corresponding CoS, but can be used with any turn. The turn line credits can be used with any class-of-service but are limited to frames using that turn value in the next node.

Both turn credits and CoS credits are subdivided into request credits and write/response credits. This is necessary to prevent a PCI deadlock condition (for more information, see Section 3.6).

3.2.2.2 Line Credit Initialization

The SG2010 allocates a number of line credits to represent its own buffers. Part of the initialization process involves providing these line credit values to its link partner(s), so that the link partners can initialize their credit counters. The SG2010 initializes its credit counters with the line credit information that the SG2010 receives from its link partners. For more information about the initialization process, including a detailed description of line credit initialization, see Section 3.8.

3.2.2.3 Reallocating Line Credits

Software may use the software generated frame mechanism to generate a special bulk line credit update frame to be sent to a link partner.This allows software to reallocate the number of line credits at the link partner's disposal for sending frames to the SG2010, or vice versa when this frame is generated at SG2010's link partner and sent to the SG2010.

The SG2010 has no credit reallocation restrictions for its line buffers. That is, software may reallocate between different CoS, or between Turn 0 and any CoS. Software should not reallocate credits to other Turn values, since Turn 0 is the only type of turn credit are used when sending frames to the SG2010. Request and write credits can never be exchanged; this is not a legal StarFabric protocol operation.

Reallocation of credits contained in the SG2010's credit counters is subject to the credit reallocation restrictions in the link partner.

The credit update frame indicates the amount by which the corresponding line credit counter(s) is to be incremented. Line credit counters may be updated at any time. It is possible to reduce the number of credits in a line credit counter by incrementing the counter by a large enough amount so that the counter wraps. The SG2010 line credit counters are seven bits wide, allowing a maximum count of 127. If a credit counter is set to 12 credits, for example, it may be incremented by 128 – 12 = 116 to reduce the line credit count to 0. Because a reduction in line credit count depends on the current value of the counter, this operation should only be performed when that particular line credit counter is not being used.

The final credit values set by software with credit update frames sent from the SG2010 to its link partner must adhere to the following rules:

- Address-routed request credits plus Turn 0 request credits must not exceed 15
- Isochronous request credits plus Turn 0 request credits must not exceed 15
- Provisioning request credits plus Turn 0 request credits must not exceed 15
- Any class of write credits, if set to a non-zero value, must have at least:
  - 13 credits if the links are bundled
  - 11 credits if the links are not bundled but the Cache Line Size is 32 dwords
  - 9 credits for all other cases

Note: *Violating this rule could result in a deadlock condition.*

3.2.2.4 Using Line Credits

In order for the SG2010 to accept incoming PCI transactions, or to initiate a PCI transaction requiring a response to be returned, the SG2010 must first have enough line credits to guarantee the resulting frame can be sent to its link partner. If the SG2010 is the target and insufficient line credits exist for that particular transaction, the SG2010 returns target retry to the PCI initiator and does not buffer the transaction. If the SG2010 is the initiator of a PCI transaction that results in a response frame (for instance, a read), it does not initiate the transaction until enough line credits exist to accommodate the response frame. The SG2010 may also have internally generated frames to send, such as software generated frames, event frames, or CSR read data. Again, the appropriate amount of line credits for the link partner must be available before the SG2010 can send these frames, even though they are already buffered or generated internally to the SG2010.

In order to prevent the starvation of larger PCI transactions by smaller transactions, the SG2010 has a minimum write/response line credit threshold that must be met when determining whether to accept a PCI transaction or initiate a read transaction. This threshold is either 5 or 9 line credits, depending on the cache line size. However, when the frame is sent, the amount corresponding to the size of the resulting frame is subtracted from the credit. Requests always require only one line to be available, and are always single line frames.

Either turn credits or CoS credits may be used to send a frame. The SG2010 does not combine line credits from the CoS and turn credit counters to come up with the total number of credits. It must have sufficient number of credits in one or the other.

The SG2010 first checks the line credits allocated to the turn value for the link partner using the first active turn in the path. If there are sufficient turn credits available, the SG2010 accepts the transaction and when the frame is sent, the SG2010 subtracts the amount of credit from that output link's line credit counter corresponding to that turn value.

If there are insufficient credits in the turn counter to accommodate the frame then the SG2010 checks the appropriate CoS credit counter for that link. If there are sufficient CoS credits then the SG2010 subtracts the amount of credit from that buffer counter.

Because address-routed frames are routed based on address decoding at each node, the correct turn is not known for the next node, and turn credits cannot be used to send address-routed frames. Therefore, the SG2010 must use CoS credits to send address-routed frames.

The following transaction types use request credits:

- All read requests
- I/O or configuration writes

All other cases use write/response credits. These cases are:

- Write frames except PCI I/O and configuration writes
- Read completion frames, all types
- Write acknowledge frames
- Provisioning frames (events, bandwidth allocation)

Special frames do not require line credits as they do not require buffer space in the link partner. Special frames are used for fabric enumeration and line credit updates between link partners.

The decision to send a frame does not solely rely on line credit information. The frame can only be sent if ordering rules allow (see Section 3.6). Also, if it has been determined that the frame has sufficient credits, it arbitrates with other frames that are also eligible to be sent to that same port (frame arbitration is described in Section 3.6.4).

3.2.2.5 Returning Line Credits to Link Partners

The SG2010 link partners use line credits to send frames to the SG2010 similar to the way the SG2010 uses line credits to send frames to the fabric. The SG2010 returns these line credits back to its link partner when the SG2010 frees buffer space that was occupied by a received frame. To return the appropriate type of line credit, the SG2010 tracks the link where the frame was received, the credit type used by the link partner (turn or CoS), and the class-of-service. If the CoS is asynchronous, high-priority isochronous, or high-priority asynchronous, then these credits are aliased to address-routed, isochronous, and provisioning, respectively, for line credit return. The Line Debit Type bit in the prepended link overhead field of the frame indicates the type of credit (turn or CoS) that was used by the link partner for that frame. If the Line Debit Type bit is 0, then CoS credits were used. If a 1, then turn credits were used. The turn credit used for incoming frames to the SG2010 is always 0.

Line credits can be returned in a line credit byte that is located in the link overhead portion of any frame containing a sequence number, with the exception of the special bulk credit update frame, that is targeted for that link partner. In the line credit byte, the SG2010 returns credits for one specific credit type.

Line credits can also be returned in a special frame that allows bulk crediting of all turn or CoS line credit types at once. There are two types of special bulk credit frames – one dedicated to returning CoS line credits (Special CoS Credit frame) and one dedicated to returning turn line credits (Special Turn Credit frame). The SG2010 uses the Special CoS Credit frame to return CoS line credits when the accumulated number of line credits reaches eight or more credits for four or more CoS categories. The SG2010 uses the Special Turn Credit frame to return turn credits when the accumulated number of line credits reaches 16 or more write credits and four or more request credits.

Additionally, the SG2010 sends a special bulk credit frame after 15 empty frames have been sent on a given link.

In both bulk credit cases, the SG2010 alternates between sending Special CoS Credit frames and Special Turn Credit frames. The alternation algorithm may force the SG2010 to send a bulk credit frame with no credits.

3.3 PCI Operation

The Bridge and the Gateway each has its own set of standard PCI configuration bits to control operation and set status. When the SG2010 Bridge function decodes or initializes a transaction on the PCI bus, Bridge configuration register bits are used. When the SG2010 Gateway function decodes or initializes a transaction on the PCI bus, Gateway configuration register bits are used. All device-specific configuration bits are located in Gateway configuration space. Some of these bits apply to the Gateway, some to the Bridge, and some to both functions.

The SG2010 supports the PCI commands shown in Table 3–6.

Table 3–6 Supported PCI Commands

Command	As Target	As Master
Special Cycle	No	Yes
Interrupt Acknowledge	No	No
I/O Read	Yes	Yes
I/O Write	Yes	Yes
Memory Write	Yes	Yes
Memory Read	Yes	Yes
Configuration Read	GW: T0 to registers Bridge: T0 to registers (Primary only) Bridge: T1 Downstream	Bridge: T1 or T0 Downstream
Configuration Write	GW: T0 to registers Bridge: T0 to registers (Primary only) Bridge: T1 Downstream Bridge: T1 Upstream (Special Cycle only)	Bridge: T1 or T0 Downstream Bridge: T1 Upstream (Special Cycle only)
Memory Read Line	Yes	Yes
Dual Address Cycle	Yes	Yes
Memory Read Multiple	Yes	Yes
Memory Write and Invalidate	Yes	Yes

The SG2010 has the following characteristics as a PCI target:

- The SG2010 responds to transactions with medium DEVSEL_L timing.
- The SG2010 only supports linear increment address mode. The SG2010 responds to other address modes as a 32-bit target and disconnects after the first data phase.
- If an address parity error is detected, the SG2010 does not respond to the transaction but signals an Address Parity Error event.

3.3.1 Memory Writes

3.3.1.1 Responding as a PCI target

Memory writes can either be directed to SG2010's CSRs through Gateway BAR0, forwarded as address-routed frames through the Bridge function, or forwarded as path-routed or multicast frames through the Gateway function. For more information about decoding addresses of PCI memory transactions, see Section 3.1.4.7.

The SG2010 accepts a memory write when it has enough buffer resources both in the SG2010 and, if the transaction is forwarded, in the next node. For a description of SG2010's buffer management, see Section 3.2.

If the initiator requests a 64-bit operation by asserting REQ64_L and if the SG2010 64-bit interface is enabled, the SG2010 asserts ACK64_L in response.

The SG2010 generates a new frame for every 128 bytes of data transferred for non-byte-enabled write and every 16 bytes for byte-enabled writes. If the SG2010 runs out of the appropriate line credits during the transfer, it returns a target disconnect to the initiator and ends the transaction. The SG2010 disconnects PCI write transactions at 4KB boundaries. The SG2010 disconnects writes to be translated into provisioning CoS frames at 4Dword boundaries.

3.3.1.2 Initiating Memory Writes as a PCI Master

When the SG2010 initiates a PCI memory write:

- It asserts REQ64_L if its 64-bit interface is enabled, the address is quadword aligned, and the SG2010 has four or more Dwords of data to deliver.
- The SG2010 performs fast back-to-back write transactions if the Fast Back-to-Back Enable bit is set in the corresponding Command register
- The SG2010 combines writes as described in Section 3.3.1.2.2.
- The SG2010 uses the Memory Write and Invalidate command as described in Section 3.3.1.2.1

Path-routed, multicast, or memory address-routed write frames received from the fabric are converted into a PCI memory write or MWI transactions. The maximum data payload of a write frame is 128 bytes (32 Dwords), although the SG2010 can combine multiple frames to create longer PCI transaction bursts.

The SG2010 master terminates a memory write transaction when:

- It delivers all the write data for that transaction and it cannot write combine with a subsequent frame.
- The master latency timer expires and the SG2010 does not have the grant (the SG2010 terminates on the next cache line boundary for MWI).
- A master abort or target abort is detected on the PCI bus.
- The SG2010 transaction arbiter has determined that this transaction should be interrupted to send another transaction. The transaction is master terminated after 32 PCI clock cycles, on the next aligned 32-Dword boundary. For a description of the transaction arbiter, see Section 3.6.3.

Note: *StarFabric protocol specifies that a frame's data payload cannot cross a 4KB addressing boundary. This must be guaranteed by the origin of the frame. As a PCI bus master, the SG2010 does not support PCI transactions that cross the 4KB boundary. The results are unpredictable if the write data payload of a received frame crosses the 4KB boundary.*

3.3.1.2.1 Initiating Memory Write and Invalidate Transactions

The SG2010 initiates a memory write transaction using the Memory Write and Invalidate command when all of the following are true:

- The Memory Write and Invalidate Enable bit is set in the Command configuration register in the

  - Bridge function if the frame was an address-routed frame.
  - Gateway function if the frame was a path-routed or multicast frame.
- The transaction starts on an aligned cache line boundary.
- At least one full cache line of data is buffered with all bytes enabled. The cache line value of the appropriate Bridge or Gateway function is used.

The SG2010 guarantees that it will begin and end an MWI transaction on a cache line boundary. If the amount of data buffered does not end on a cache line boundary, the SG2010 terminates the transaction on the last cache line boundary and delivers the remaining write data with a transaction using the Memory Write command.

When the SG2010 is the master of a memory write, it does not know whether the initiator used a Memory Write or an MWI command. The SG2010 determines whether to use an MWI command based solely on the properties of the data as described above.

3.3.1.2.2 Write Combining

The SG2010 can be enabled to combine writes for frames that are translated into PCI memory write transactions. The SG2010 combines multiple write frames into a single PCI transaction if the frames meet the following criteria:

- The subsequent PCI memory address is consecutive and contiguous to the previous PCI memory address. There can be no address holes.
- The write frames belong to the same class-of-service.
- The write frames use the write (without acknowledge) operation.
- The Write Combine Enable bit in the Gateway Chip Control configuration register is set. This bit applies to transactions initiated by either the Bridge or Gateway function.

Write combining is not performed when:

- A frame has reached or crossed a 4KB boundary
- A frame has reached or crossed an 128 byte boundary and there is a pending delayed request transaction
- One write frame is byte-enabled and the other is not byte-enabled
- A frame is a memory write with acknowledge
- A frame is a memory write with error (contains parity errors)

3.3.1.3 PCI Errors During Writes

3.3.1.3.1 PCI Errors on the Initiator Bus

If the SG2010 detects a parity error when receiving write data, it sets the Detected Parity Error bit in the Bridge or Gateway Status register, depending on which function decoded the transaction. The SG2010 asserts PERR_L if the Parity Error Response (PER) bit is set in the corresponding Command register. The SG2010 also sets the PCI Status: Detected Parity Error chip event bit.

If the Parity Error Response bit is not set and a parity error is detected, the SG2010 continues the transaction and sends the write data into the fabric without any error or event indication.

If the Parity Error Response bit is set and the SG2010 has already received at least one Dword of good data during the transaction when a parity error is detected, the SG2010 disconnects the transaction. This disconnect is performed for both Memory Write commands and Memory Write and Invalidate commands, and enables the data with parity error to be sent in a frame separate from the good data.

If the Parity Error Response bit is set and the SG2010 detects a parity error during the first data phase, the SG2010 continues to accept write data until a data phase with good parity is received. The SG2010 disconnects the transaction after the next data phase so that the good data can be sent in a frame separate from the bad data.

If the Parity Error Response bit is set and the SG2010 detects a data parity error on a write to be translated into a provisioning CoS frame, then the SG2010 sends the resulting frame with all data bytes disabled; that is, no data is written at the target. The provisioning CoS does not allow a write with error operation. The SG2010 signals the Parity Error on Provisioning Write event.

3.3.1.3.2 PCI Errors on the Target Bus

When the SG2010 initiates a write transaction on the PCI bus, the transaction can encounter the following errors: data parity error, master abort, target abort, and target response time-out. PCI status bits are set:

- in the Bridge function’s Status configuration register if the transaction is initiated by the Bridge function (that it, it was translated from an address-routed frame), and the PCI bus is the primary bus
- in the Bridge function’s Secondary Status configuration register if the transaction is initiated by the Bridge function (that it, it was translated from an address-routed frame), and the PCI bus is the secondary bus
- or the Gateway function’s Status register if the transaction is initiated by the Gateway function (it was not translated from an address-routed frame).

3.3.1.3.2.1 Data Parity Error

If the SG2010 detects PERR_L asserted when it is delivering write data and if its Parity Error Response bit is set, it sets the Master Data Parity Error status bit in the appropriate Status register. The SG2010 also sets the PCI Status: Master Data Parity Error chip event bit. If a write acknowledge frame is returned, the Failure Type is set to Parity Error.

3.3.1.3.2.2 Master Abort

If the SG2010 receives a master abort in response to a memory write, the SG2010 discards the write data and sets the Master Abort Received bit in the appropriate Status register. The SG2010 also sets the PCI Status: Master Abort Received chip event bit. If a write acknowledge frame is returned, the Failure Type is set to Master Abort.

3.3.1.3.2.3 Target Abort

If the SG2010 receives a target abort in response to a memory write, the SG2010 discards the remaining write data and sets the Target Abort Received bit in the appropriate Status register. The SG2010 also sets the PCI Status: Target Abort Received chip event bit. If a write acknowledge frame is returned, the Failure Type is set to Target Abort.

3.3.1.3.2.4 Target Response Time-out

The SG2010 implements a target response timer to monitor the progress of transactions that the SG2010 initiates on the PCI bus. When the SG2010 initiates a memory write transaction (receives a PCI grant and asserts FRAME_L), the timer begins counting on each PCI clock cycle. If any data is transferred, or the transaction terminates due to a master abort or target abort, then the timer resets to 0. If the timer expires, then the SG2010 discards the write transaction. Timer expiration occurs when 2^{26} PCI clock cycles have passed and the timer has not been reset. The SG2010 sets the Target Response Timer Expired chip event on timer expiration. If a write acknowledge frame is returned, the Failure Type is set to Time-out.

3.3.2 Memory Reads

The SG2010 treats PCI memory reads as delayed transactions. When the SG2010 detects a memory read it queues the read information and returns a target retry. The SG2010 continues to return target retry to that read transaction until read completion status is available. The SG2010 has eight PCI Delayed Transaction Buffer entries that are used to buffer incoming delayed transactions.

Memory reads can either be directed to the SG2010 CSRs, forwarded as address-routed frames using the SG2010 Bridge function, or forwarded as path-routed frames using the SG2010 Gateway function. For more information about decoding addresses of PCI memory transactions, see Section 3.1.4.7.

3.3.2.1 Prefetchable and Non-Prefetchable Reads

A non-prefetchable read transaction uses the Memory Read command and is directed to non-prefetchable memory space. All other types of memory reads (Memory Read to prefetchable space, Memory Read Line, Memory Read Multiple) are considered to be prefetchable.

The SG2010 non-prefetchable space includes:

- Any address decoded by the Bridge function’s 32-bit memory range defined by the Memory Base and Memory Limit registers
- If the Prefetch Disable bit is set in the Gateway’s Chip Control configuration register, any memory address inversely decoded by the Bridge function
- Any address decoded by the Gateway BAR2, BAR3, BAR4, and BAR5 ranges, if the BAR is selected to be non-prefetchable through the BAR’s corresponding setup register (individually selectable)

When the SG2010 decodes a non-prefetchable read as a target, it sets the number of Dwords requested to 1 in the corresponding read request frame. The SG2010 also captures the byte enables and includes them in the frame. If the read is to the SG2010 CSRs, only a single Dword is returned.

For the prefetchable cases, the SG2010 sets the number of Dwords requested as dictated by the bus commands and the programmable values in the Chip Control register in Gateway configuration space. The SG2010 never requests a read amount that crosses a 4KB boundary. If the prefetchable read amount specified would cause the read to cross a 4KB boundary, then the SG2010 adjusts the Dwords requested in the frame to go up to, but not cross, the 4KB address boundary.

When the SG2010 forwards a prefetchable read, all byte enables are forced to zero; that is, all bytes are enabled.

Note: *The SG2010 does not support PCI transactions crossing the 4KB boundary as a PCI bus master. StarFabric protocol restricts Dwords requested from crossing a 4KB addressing boundary. If a frame is received that requests a data payload that crosses a 4KB boundary, results are unpredictable.*

3.3.2.2 Queueing Memory Reads from the PCI Bus

The SG2010 queues a memory read into the PCI delayed Transaction Buffer when:

- The transaction is to be converted into a frame and forwarded to the fabric (not a CSR read)
- A transaction using that address is not already queued in the buffer
- Enough buffer space is available in the SG2010 and the next node (See Section 3.2)
- The appropriate type of PCI delayed transaction entry is available

When a read is queued, the SG2010 generates either an address-routed PCI read request frame or a path-routed read request frame, depending on the function that decodes the PCI transaction.

If the read is speculative, the amount of data requested by the frame is based on the cache line size, the memory command used, and device-specific prefetch control bits ([5:4], [7:6], [9:8]) in the Gateway configuration Chip Control register (described in Section 4.8.8.1). Table 3–7 summarizes these amounts. Memory Read command prefetch amounts apply only if the read is prefetchable; otherwise, 1 Dword is requested. Memory Read Line and Memory Read Multiple commands always use these prefetchable amounts.

Table 3–7 Speculative Read Prefetch Amounts

	Memory Read Prefetch Bits Value			
Read Command	00b	01b	10b	11b
Memory Read (pf)	1 cache line	2 cache lines	4 cache lines	1 Dword
Memory Read Line	1 cache line	2 cache lines	4 cache lines	8 cache lines
Memory Read Multiple	2 cache lines	4 cache lines	8 cache lines	16 cache lines

A prescriptive read is generated through a source channel enable bit. The Source Channel Table also contains the amount of Dwords to be requested for a prescriptive read, which can range from 1 to 512 Dwords, in single Dword granularity. The SG2010 generates path-routed prescriptive reads only.

The SG2010 does not request a read amount that crosses a 4KB boundary. If the prescriptive read amount specified in the Source Channel Table would cause the read to cross a 4KB boundary, then the SG2010 adjusts the prescriptive Dwords requested in the frame to go up to, but not cross, the 4KB address boundary.

3.3.2.3 Initiating Memory Reads on the PCI Bus

Incoming read request frames are translated and stored in the Transaction buffer and the Write/Address buffer. When ordering rules allow the transaction to be initiated on the PCI bus, the transaction is removed from those buffers and stored in the Outstanding Transaction buffer. The eight-entry Outstanding Transaction buffer holds read, I/O write, and configuration write frames that are ready to be directed to the PCI bus. The Transaction Buffer Mode bit in the Gateway Chip Control configuration register specifies how these buffer entries are managed by the SG2010. If a CoS reservation mechanism is enabled with this bit, three entries are dedicated to specific classes-of-service – one entry for address-routed/asynchronous, one for isochronous/HP-Isochronous, and one for HP-asynchronous/provisioning. Five of those entries can be used for any CoS. If the CoS reservation mechanism is disabled, all eight entries can be used for any CoS.

When a transaction is queued in the Outstanding Transaction buffer, the SG2010 initiates a memory read transaction on the PCI bus, unless there is insufficient space in the Output Buffer to hold the resulting read data.

The SG2010 chooses the read command based on the amount of read data requested by the frame. If the requested amount is:

- 1 Dword, the memory read command is used
- Up to a cache line of data, the memory read line command is used
- More than a cache line of data, the memory read multiple command is used.

Up to 2KB may be requested.

The SG2010 creates a new read completion frame for every 128 bytes of data received from the target. The SG2010 continues the read until it receives the entire amount requested, the target disconnects, a master latency time-out condition exists, or a parity error is detected. If the transaction is terminated with error, the SG2010 does not request any more read data from the target for a given read request. If the transaction is terminated without error and the read is prescriptive, the SG2010 may initiate another read transaction as described in Section 3.3.2.4.

3.3.2.4 Speculative and Prescriptive Reads

The SG2010 supports both speculative and prescriptive reads.

If the SG2010 receives a speculative read request frame with a number of Dwords requested, the SG2010 initiates a single PCI read transaction and attempts to read the entire amount requested. If the transaction terminates either due to target termination or master latency timer expiration, the SG2010 does not attempt to read any more data.

If the SG2010 receives a prescriptive read request frame with a number of Dwords requested, and the transaction is terminated due to a disconnect or master latency timer expiration before the total amount is read, then the SG2010 initiates another read transaction to continue to acquire more read data. The SG2010 initiates as many transactions as needed to read the entire amount requested, unless a master abort, target abort, or target response error is encountered. As a PCI bus master, the SG2010 supports prescriptive reads from 1 to 512 Dwords, with single Dword granularity.

3.3.2.5 Returning Read Data to the PCI Initiator

The SG2010 returns read data to the PCI initiator as soon as it has some read data available and as soon as ordering rules allow. The SG2010 does not order return read data with other read completions; that is, read completions can be returned in any order with respect to each other.

The SG2010 continues to return read data regardless of frame boundaries – that is, the SG2010 can receive subsequent completion frames for that read and continue the same transaction. Read data is returned until the master terminates the transaction, or until all the data currently buffered in response to that request is consumed. When the initiator consumes the buffered read data, the SG2010 disconnects the read transaction.

After the transaction has been terminated, any subsequent read completion frames corresponding to that read request are discarded (if read retention is not enabled or the read is not prescriptive).

3.3.2.5.1 Read Data Retention

If a read transaction is terminated and more read data remains (or is buffered by subsequent completion frames in that sequence), the SG2010 can retain the read data in its buffers. If the read was prescriptive, the SG2010 always retains data. If the read was speculative, the Read Data Retention Enable bit must be set in the Gateway Chip Control configuration register.

When the SG2010 is retaining read data and if a read transaction is initiated containing the address of the first retained Dword, then the SG2010 returns the retained data to the initiator. The SG2010 continues to retain data on any subsequent master termination as long as it has more read data remaining.

A retained read data transaction not only consumes read completion buffering, but also continues to occupy an entry in the PCI Delayed Transaction buffer. A speculative retained read data transaction is discarded when the PCI Delayed Transaction buffer is full and an initiator attempts a delayed transaction that could be buffered by the SG2010. Any type of retained read transaction is discarded when a Master Time-out condition exists. The Master Time-out counter resets every time the SG2010 transfers retained read data to the initiator.

3.3.2.6 PCI Errors during Memory Reads

When the SG2010 initiates a read transaction on the PCI bus, the transaction can encounter the following errors: data parity error, master abort, target abort, and target response time-out. PCI status bits are set:

- in the Bridge function’s Status configuration register if the transaction is initiated by the Bridge function (that it, it was translated from an address-routed frame), and the PCI bus is the primary bus
- in the Bridge function’s Secondary Status configuration register if the transaction is initiated by the Bridge function (that it, it was translated from an address-routed frame), and the PCI bus is the secondary bus
- in the Gateway function’s Status register if the transaction is initiated by the Gateway function (it was not translated from an address-routed frame).

3.3.2.6.1 PCI Read Errors on the Target Bus

3.3.2.6.1.1 Master Abort

If the SG2010 receives a master abort in response to a PCI read transaction, the SG2010 sets the Received Master Abort bit in the appropriate Status register. The SG2010 does not attempt to read any more data, even if the read is a prescriptive read. The Master Abort failure type is indicated in the read completion frame. The SG2010 also sets the PCI Status: Master Abort Received chip event bit.

3.3.2.6.1.2 Target Abort

If the SG2010 receives a target abort in response to a PCI read transaction, the SG2010 sets the Received Target Abort bit in the appropriate Status register. The SG2010 does not attempt to read any more data, even if the read is prescriptive. The Target Abort failure type is indicated in the read completion frame. The transaction is terminated on the initiator PCI bus according to PCI protocol on the data phase where the target abort occurs. The SG2010 also sets the PCI Status: Target Abort Received chip event bit.

3.3.2.6.1.3 Data Parity Error

If a data parity error occurs when reading data from a target, the SG2010 sets the Detected Parity Error bit in the appropriate Status register. If the Parity Error Response bit is set the SG2010 sets the Master Data Parity Error bit in that same Status register. The SG2010 also set the PCI Status: Detected Parity Error and, if the Parity Error Response bit is set, the PCI Status: Master Data Parity Error chip event bits.

When a change of parity state is detected (from good to bad or bad to good), and the Parity Error Response bit is set, the SG2010 master terminates the transaction as soon as possible. The data read is put into a Read Completion frame with the operation specified as Read Completion with Error. If the read is prescriptive, the SG2010 initiates another transaction to continue to obtain the remaining read data.

If all of the data read has bad parity, the SG2010 does not master terminate the transaction, but the Read Completion frame containing the data specifies the operation as Read Completion with Error.

3.3.2.6.1.4 Target Response Time-out

The SG2010 implements a target response timer to monitor the progress of transactions that the SG2010 initiates on the PCI bus. When the SG2010 initiates a memory read transaction, the timer begins counting on each PCI clock cycle. If any data is transferred, or the transaction terminates due to a master abort or target abort, then the timer resets to 0. If the timer expires, then the SG2010 discards the read transaction and returns a read completion frame with the Time-out failure status. Timer expiration occurs when 2^{26} PCI clock cycles have passed and the timer has not been reset. The SG2010 also sets the PCI Status: Target Response Time-out event bit.

3.3.2.6.2 PCI Read Errors on the Initiator Bus

3.3.2.6.2.1 Target Abort

If the SG2010 receives a read completion frame indicating a target abort failure type, the SG2010 returns target abort to the initiator during the same data cycle as it was originally detected. The SG2010 sets the Signaled Target Abort bit in the appropriate Status register. The SG2010 also sets the PCI Status: Signaled Target Abort chip event bit.

3.3.2.6.2.2 Master Abort

If the SG2010 receives a read completion frame with a Master Abort, Hardware Distribution Failure, or Software Distribution Failure failure type, the SG2010 uses the Master Abort Mode bit in the Bridge Control register and the PCI Target Response Mode bit in the Gateway's Chip Control register to determine its response to the initiator. If neither bit is set, the SG2010 returns FFFF FFFFh as data to the master and disconnects after the first data phase. If either or both bits are set, the SG2010 returns a target abort to the master, sets the Signaled Target Abort bit in the appropriate Status register, and sets the PCI Status: Signaled Target Abort chip event bit.

If the SG2010 receives a prescriptive read completion frame with a Failure Type of Master Abort, Hardware Distribution Failure, or Software Distribution, and it is not the first completion frame for that transaction, then the SG2010 returns target abort to the PCI initiator regardless of the Master Abort mode bit value.

3.3.2.6.2.3 Parity Error

If the SG2010 receives a read completion frame with the read completion with error command, the SG2010 forces bad parity for those data phases.

3.3.2.6.2.4 Target Response Time-out

If the SG2010 receives a read completion frame with the Time-out failure type set, it returns a target abort to the PCI initiator of the read transaction and sets the Signaled Target Abort bit in the appropriate Status register. The SG2010 also sets the PCI Status: Signaled Target Abort chip event bit.

3.3.2.6.2.5 Response Frame Time-out

The SG2010 implements a 2^{32} cycle response frame timer for each delayed transaction. The timer starts when the SG2010 sends a read request frame into the fabric and counts once per PCI clock cycle. The timer resets when a response frame is received for that transaction. If it is not the last response frame, the timer begins counting again.

If the response frame timer for an outstanding read transaction expires, then the SG2010 returns a target abort in response to the PCI read transaction, and sets the Response Frame Timer Expired chip event bit.

3.3.2.6.2.6 Master Time-out

The SG2010 Master Time-out Counter starts counting once the SG2010 has read data ready to return to the master. The Master Time-out Counter is initialized to either 2^{10} or 2^{15} (default) PCI clock cycles. If the counter for a particular read transaction expires before the SG2010 detects that the master has reattempted the delayed transaction, the SG2010 discards the read transaction. The SG2010 sets the Master Time-out status bit, and the PCI Status: Delayed Transaction Master Time-out chip event bit.

3.3.3 I/O Writes

An I/O write is treated as a delayed transaction on the PCI interface. The SG2010 has eight initiator transaction entries used to buffer incoming delayed transactions.

3.3.3.1 Queueing I/O Writes

The SG2010 queues an I/O write into the Initiator Transaction buffer when:

- The transaction is not directed to SG2010’s CSRs
- A transaction using that address is not already queued
- Enough buffer space in the SG2010 and the link partner is available (see Section 3.2).
- An appropriate initiator transaction entry is available

When an I/O write is queued, the SG2010 generates an address-routed I/O write frame as described in Section 3.4.2.

3.3.3.2 Initiating I/O Writes

Similar to incoming read request frames, incoming I/O write frames are translated and stored in the Transaction buffer and the Write/Address buffer. When ordering rules allow the transaction to be initiated on the PCI bus, the transaction is removed from those buffers and stored in the Outstanding Transaction buffer. (This buffer is described in Section 3.3.2.3.) The SG2010 initiates an I/O write on the PCI bus if there is enough space in the Output Buffer to hold the response. Otherwise the SG2010 waits to initiate the transaction until enough space is available.

When the SG2010 initiates the I/O write, it receives the target response and encodes it into a Failure Type field in a write acknowledge frame.

3.3.3.3 Returning I/O Write Status

The SG2010 returns write status to the PCI initiator as soon as ordering rules allow. The SG2010 disconnects all I/O writes after a single Dword is transferred from the initiator. Write status can be TRDY_L (data written with no errors), TRDY_L and PERR_L (data written but parity error detected), or target abort. A target abort can mean that write data was not delivered on the target bus due to a target abort, a master abort, or another error such as a retry time-out. For a target termination summary, see Table 3–8.

3.3.3.4 PCI Errors during I/O Writes

When the SG2010 initiates an I/O write transaction on the PCI bus, the transaction can encounter the following errors: data parity error, master abort, target abort, and target response time-out. PCI status bits are set:

- in the Bridge function’s Status configuration register if the transaction is initiated by the Bridge function (that it, it was translated from an address-routed frame), and the PCI bus is the primary bus
- in the Bridge function’s Secondary Status configuration register if the transaction is initiated by the Bridge function (that it, it was translated from an address-routed frame), and the PCI bus is the secondary bus

3.3.3.4.1 PCI I/O Write Errors on the Target Bus

3.3.3.4.1.1 Master Abort

If the SG2010 receives a master abort in response to a PCI I/O write transaction, the SG2010 sets the Received Master Abort bit in the appropriate Status register. The Master Abort failure type is indicated in the write acknowledge frame. The SG2010 also sets the PCI Status: Master Abort Received chip event bit.

3.3.3.4.1.2 Target Abort

If the SG2010 receives a target abort in response to a PCI I/O write transaction, the SG2010 sets the Received Target Abort bit in the appropriate Status register. The Target Abort failure type is indicated in the write acknowledge frame. The SG2010 also sets the PCI Status: Received Target Abort chip event bit.

3.3.3.4.1.3 Data Parity Error

If a data parity error occurs (PERR_L is asserted) when writing data to the target, the SG2010 sets the Detected Parity Error in the appropriate Status register. If the Parity Error Response bit is set the SG2010 also sets the Master Data Parity Error bit in the same Status register. The SG2010 also sets the PCI Status: Detected Parity Error chip event bit and, if the Parity Error Response bit is set, the PCI Status: Master Data Parity Error chip event bit.

3.3.3.4.1.4 Target Response Time-out

The SG2010 implements a target response timer to monitor the progress of transactions that the SG2010 initiates on the PCI bus. When the SG2010 initiates an I/O write transaction, the timer begins counting on each PCI clock cycle. If data is transferred, or the transaction terminates due to a master abort or target abort, then the timer resets to 0. If the timer expires, then the SG2010 discards the I/O write transaction and returns a write acknowledge frame with the time-out failure status. Timer expiration occurs when 2^{26} PCI clock cycles have passed and the timer has not been reset. The SG2010 set the Target Response Timer Expired chip event bit.

3.3.3.4.2 PCI I/O Write Errors on the Initiator Bus

3.3.3.4.2.1 Target Abort

If the SG2010 receives a write acknowledge frame indicating a target abort failure type, the SG2010 returns target abort to the initiator. The SG2010 sets the Signaled Target Abort bit in the appropriate Status register. The SG2010 also sets the PCI Status: Signaled Target Abort chip event bit.

3.3.3.4.2.2 Master Abort

If the SG2010 receives a write acknowledge frame with a Master Abort, Hardware Routing Failure, or Software Routing Failure failure type, the SG2010 uses the Master Abort Mode bit in the Bridge Control register and the PCI Target Response Mode bit in the Gateway's Chip Control register to determine its response to the initiator. If neither bit is set, the SG2010 returns TRDY_L to the master. If either or both of the bits are set, the SG2010 returns a target abort to the master and sets the Signaled Target Abort bit in the Bridge Status register corresponding to the PCI interface and sets the PCI Status: Signaled Target Abort chip event.

3.3.3.4.2.3 Parity Error

There are two cases of parity error reporting on the initiator bus. The first case is when the parity error occurs and is detected on the initiating bus. The second is when the parity error occurred on the target bus and the parity error failure type is returned in a write acknowledge frame.

In the first case, if the SG2010 detects a parity error on the initiator bus, and the Parity Error Response bit for that function and interface is set, the SG2010 asserts TRDY_L and discards the write. The write is not queued. If the Parity Error Response bit is not set, the transaction is queued normally. The SG2010 sets the Detected Parity Error bit in the appropriate Status register and sets the PCI Status: Detected Parity Error chip event bit.

If the SG2010 receives a write acknowledge frame with the parity error failure type, and the Parity Error Response bit is set, the SG2010 asserts PERR_L to indicate a parity error in the first data phase. The SG2010 sets the Detected Parity Error bit in the appropriate Status register and sets the PCI Status: Detected Parity Error chip event bit.

3.3.3.4.2.4 Response Frame Time-out

The SG2010 implements a 2^{32} clock cycle response frame timer for each outstanding delayed transaction. The timer starts when the SG2010 sends an I/O write frame into the fabric and counts once per PCI clock cycle. The timer resets when a response frame is received for that transaction. If it is not the last response frame, the timer begins counting again.

If the response frame timer for an outstanding I/O write transaction expires, then the SG2010 returns a target abort in response to the I/O write transaction, and sets the Response Frame Timer Expired chip event bit.

3.3.3.4.2.5 Master Time-out

The Master Time-out Counter starts counting once the SG2010 has write status ready to return to the master. The Master Time-out Counter is initialized to either 2^{10} or 2^{15} (default) PCI clock cycles. If the counter expires before the SG2010 detects that the master has re-attempted the delayed transaction, the SG2010 discards the transaction. The SG2010 sets the Master Time-out status bit, and the PCI Status: Delayed Transaction Master Time-out chip event bit.

3.3.4 I/O Reads

An I/O read is treated as a delayed transaction on the PCI interface. The SG2010 uses the target transaction buffer to hold I/O reads moving from the fabric to the PCI bus. This buffer is described Section 3.3.2.3.

I/O reads are treated the same way as non-prefetchable memory reads by the SG2010 as both a target and initiator on the PCI bus. I/O reads are restricted to a single, byte-enabled Dword. There are no prescriptive I/O reads, nor is read data retained (since only a single Dword is requested). Errors are handled in the same way as for memory reads.

3.3.5 Configuration Writes

The PCI protocol for responding to and initiating configuration writes is identical to that of I/O writes. Configuration address decoding is described in Section 3.1.4.6. Configuration frame translation is described in Sections 3.5.1.3 and 3.5.3.2.

3.3.6 Configuration Reads

The PCI protocol for responding to and initiating configuration reads is identical to that of I/O reads and non-prefetchable memory reads. Configuration address decoding is described in Section 3.1.4.6. Configuration frame translation is described in Sections 3.5.1.3 and 3.5.3.2.

3.3.7 PCI Target Termination Summary

The previous sections describe the results of PCI errors on the PCI bus. This section provides a summary of target terminations based on situations that occur both in the SG2010 as it decodes an incoming transaction, and when the SG2010 received a response frame with a given failure type. Table 3–8 summarizes the target responses that the SG2010 returns in response to these situations.

Table 3–8 PCI Target Termination Summary

Condition	Mode	Memory Write	Memory Read	I/O or Cfg Write	I/O or Cfg Read
Errors Detected at SG2010					
Data Parity Error	PER = 0	TRDY_L - queued	N/A	TRDY_L - queued	N/A
	PER = 1	TRDY_L - queued		TRDY_L - discard	
Segment Table Invalid	MAB = 0*	TRDY_L - discard	TRDY_L and FFFFFFFFh	N/A	N/A
	MAB = 1†	Target abort	Target Abort		
SG2010 Port Down	MAB = 0	TRDY_L - discard	TRDY_L and FFFFFFFFh	N/A	N/A
	MAB = 1	Target abort	Target Abort		
Invalid Multicast Group	MAB = 0	TRDY_L - discard	TRDY_L and FFFFFFFFh	N/A	N/A
	MAB = 1	Target abort	Target Abort		
Source Channel Range Error	N/A	TRDY_L - discard	Target Abort	N/A	N/A
Port Map Decode Miss	MAB = 0	TRDY_L - discard	TRDY_L and FFFFFFFFh	TRDY_L - discard	TRDY_L and FFFFFFFFh
	MAB = 1	Target abort	Target Abort	Target Abort	Target Abort
Response Frame Time-out	N/A	N/A	Target Abort	Target Abort	Target Abort
Response Frame Failure Types					
Master Abort Failure Type	MAB = 0	N/A	TRDY_L and FFFFFFFFh	TRDY_L	TRDY_L and FFFFFFFFh
	MAB = 1	N/A	Target Abort	Target Abort	Target Abort
	≥2 frame pres. rd.‡	N/A	Target Abort	N/A	N/A
Target Abort Failure Type	N/A	N/A	Target Abort	Target Abort	Target Abort
Time-out Failure Type	N/A	N/A	Target Abort	Target Abort	Target Abort

Table 3–8 PCI Target Termination Summary *(Continued)*

Condition	Mode	Memory Write	Memory Read	I/O or Cfg Write	I/O or Cfg Read
Lockout Failure Type	MAB = 0	N/A	TRDY_L and FFFFFFFFh	TRDY_L	TRDY_L and FFFFFFFFh
	MAB = 1	N/A	Target Abort	Target Abort	Target Abort
Range Failure Type	N/A	N/A	Target Abort	N/A	N/A
Channel Lock Failure Type	N/A	N/A	Target Abort	N/A	N/A
Channel Inactive Failure Type	N/A	N/A	Target Abort	N/A	N/A
Parity Error Failure Type	N/A	N/A	N/A	TRDY_L	N/A
Hardware Routing Failure	MAB = 0	N/A	TRDY_L and FFFFFFFFh	TRDY_L	TRDY_L and FFFFFFFFh
	MAB = 1	N/A	Target Abort	Target Abort	Target Abort
Software Routing Failure	MAB = 0	N/A	TRDY_L and FFFFFFFFh	TRDY_L	TRDY_L and FFFFFFFFh
	MAB = 1	N/A	Target Abort	Target Abort	Target Abort
Reserved Failure Type	N/A	N/A	Target Abort	Target Abort	Target Abort

* MAB =0: Neither of these bits are set: the Master Abort Mode bit in the Bridge function’s Bridge Control register or the PCI Target Response Mode bit in the Gateway’s Chip Control register

† MAB =1: Either or both of these bits are set: the Master Abort Mode bit in the Bridge function’s Bridge Control register or the PCI Target Response Mode bit in the Gateway’s Chip Control register

‡ Not the first frame of a prescriptive read.

3.4 PCI to Frame Translation

To move data between the StarFabric links and the PCI bus, the SG2010 translates PCI transactions into StarFabric frames and frames to PCI transactions. The *StarFabric Architecture Specification* provides information on StarFabric protocol, and the *PCI Local Bus Specification, Rev 2.2* provides information on PCI bus protocol. This section discusses the translation of the following PCI transactions into frames:

- PCI memory writes (SG2010 as PCI target)
- PCI I/O and configuration writes (SG2010 as PCI target)
- PCI write responses (SG2010 as PCI initiator)
- PCI memory reads (SG2010 as PCI target)
- PCI I/O and configuration reads (SG2010 as PCI target)
- PCI read completions (SG2010 as PCI initiator)

This section does not describe the generation of other types of frames, such as special frames, bandwidth reservation frames, or event frames. The SG2010 can generate bandwidth reservation frames and special frames using the software generated frame function, and event frames through its event dispatcher.

3.4.1 Translating PCI Memory Write Transactions

When the SG2010 is the target of a PCI memory write that is to be forwarded to the fabric, it translates the write transaction into one or more StarFabric frames. PCI memory write transactions have several parameters that must be captured when translating the transaction into StarFabric format, as shown in Table 3–9.

Table 3–9 PCI Memory Write Translation Parameters

Parameter/Condition	Values	Comment
Bus Command	Memory Write	Write operation used for both
	Memory Write and Invalidate	
Byte Enables	One or more bytes disabled	Byte masked frame used
	All bytes enabled	Byte masked frame not used
Amount of data	Single Dword	Single line frame generated
	Multiple Dwords	Multiple line frame or multiple frames required
	Amount is four Dword multiple	No orphan byte count
	Amount is not four Dword multiple	Orphan byte count required
Type of address hit	Bridge Base/Limit	Address-routed frame generated
	Gateway BAR2–BAR5	Path-routed or multicast frame generated
Parity Error on PCI bus?	No	Write operation
	Yes	Write with error operation

3.4.1.1 Generating Address-Routed Write Frame Headers

When the SG2010 generates an address-routed memory write frame, corresponding frame field values are assigned as shown in Table 3–10.

Table 3–10 Address-Routed Write Frame Field Values

Frame Field	Value	Field Width	Bit Value
Class-of-Service	Address routed	3	4h
Operation	Write (if no parity error) Write w/ Error (if parity error)	3	4h 5h
Relaxed ordering	No	1	0b
Target Region	SAC DAC	2	00b 01b
Transaction Number	Null transaction number	6	63 (3Fh)

The SG2010 does not generate address-routed write frames using the Write with Acknowledge operation through this translation mechanism.

3.4.1.1.1 Address Fields

When an address-routed write frame is generated, its path is unknown and is initialized to all 0's. The path is built dynamically as the frame travels through the fabric to the terminus. The path transform of this constructed path is used for response frames or path event frames that are returned to the origin.

PCI address bits [49:2] are included in the frame as the Offset field. If the address is a 32-bit address, then offset bits [49:32] are 0. For information about 64-bit PCI address support, see Section 3.1.4.7.6.

3.4.1.2 Generating Path-Routed and Multicast Write Frame Headers

The Segment Table, Path Table, and optionally the Source Channel Table, define the values of many of the path-routed and multicast frame header fields. If an path-routed or multicast write frame is generated, corresponding frame field values are assigned as shown in Table 3–11.

Table 3–11 Path-routed/Multicast Write Frame Field Values

Frame Field	Value	Field Width	Bit Value	Defined By
Class of Service	Asynchronous Isochronous HP-Asynchronous Multicast HP-Isochronous Provisioning	3	0h 1h 2h 3h 5h 6h	Segment Table
Operation	Write WwA[*] Write with Error WwA with Error	3	4h 6h 5h 7h	– Source Channel Table Data parity error Source Channel Table and data parity error
Relaxed ordering	No	1	0b	Fixed
Path/Multicast ID	Look-up	24	Lookup	Segment/Path Tables
Offset	–	42	–	PCI offset with MSB replacement and source channel translation
Transaction Number	Null transaction number WwA transaction number	6	63 (3Fh) 61 (3Dh)	If not WwA If WwA

* WwA = Write with Acknowledge

3.4.1.3 Data Payload in Write Frames

Although the addressing modes of path-routed, multicast and address-routed write frames are different, the data-related fields and data organization are identical.

The minimum frame size is one line, which is four Dwords (16 bytes). All frames are aligned to a line boundary. The header for a write frame occupies three Dwords of the first line. The fourth Dword in the first line may contain data, depending on the data organization. Data organization is dependent on the number of Dwords in the payload and whether internal or external overhead is used.

3.4.1.3.1 Link Overhead

Link overhead is not part of the frame, but is always sent with a frame to ensure the integrity of the transmission and to pass credit information between link partners. The link overhead for a frame consists of the frame sequence number, the CRC, and the buffer credit byte. Link overhead is four bytes. The frame sequence number byte is prepended to the frame; the remaining bytes are appended to the frame.

When there are four or more unused data bytes in the last line of the data payload, the protocol requires frame size optimization. Instead of the link overhead existing outside of the 16-byte aligned frame, the SG2010 pulls the link overhead within the frame boundary where it takes the place of four of the unused data bytes. This decreases the bandwidth consumption of the frame. In these cases, the Internal Link Overhead header bit is set to 1. In terms of how it looks in the serial transmission, a frame with internal link overhead looks like a frame with external link overhead, except that the last four empty bytes are not transmitted and the appended bytes of link overhead are moved up accordingly. For more information about link overhead and frame formats, see the *Star-Fabric Architecture Specification.*

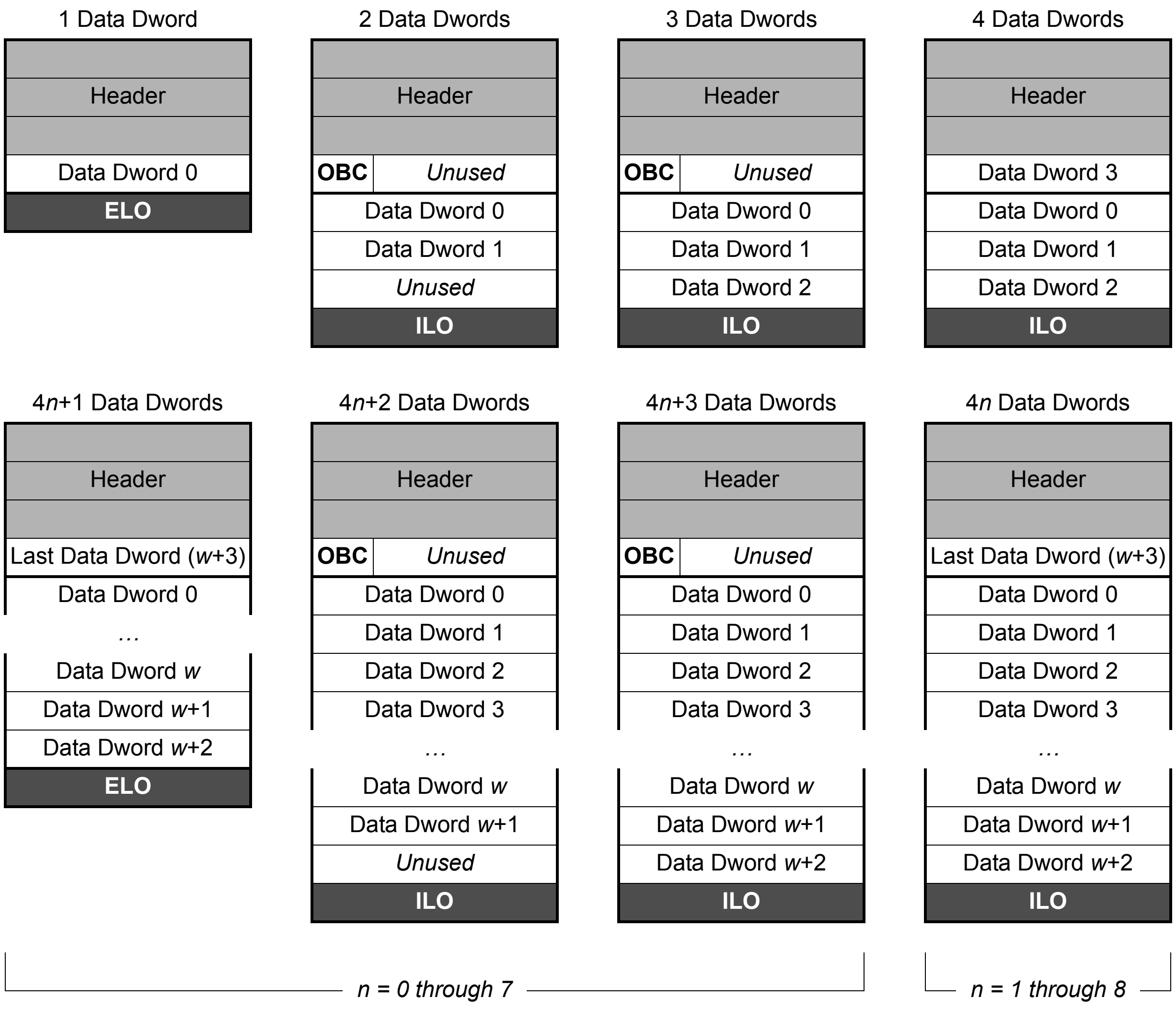
3.4.1.3.2 Data Organization

If the data payload is a multiple of four Dwords, or a multiple of four Dwords plus 1 (1, 5, 9, 13…) then the SG2010 places the last Dword of payload data in the fourth Dword location of the first line (see Figure 3–9). This arrangement allows frame size optimization. When the payload is a multiple of four Dwords, the Dword location at the end of the last line is empty, and can be occupied by internal link overhead (ILO). If the payload is a multiple of four Dwords plus 1, then the last Dword, which otherwise would be the only Dword in a line, is moved into the header. The remaining payload, if any, starts in order from the least significant Dword at the second line. All data locations in the frame are then occupied, and the link overhead is external (ELO).

If the data payload is a multiple of four Dwords plus 2 (2, 6, 10, …) or plus 3 (3, 7, 11, …) the last Dword in the header is left empty. The payload starts at the second line, and the last line of the frame contains two or three Dwords of data. Because there are two or one empty Dword locations in the last line of the frame, the link overhead is internal.

Figure 3–9 shows the frame organization for all combinations data payloads, external link overhead (ELO), internal link overhead (ILO), orphan byte count (OBC), and unused Dwords.

Figure 3–9 Frame Data Payload Organization



3.4.1.3.3 Data-related Header Fields

The remaining header fields depend on the amount and type of data to be sent.

A four-bit Additional Frame Size field indicates how many lines of data follow the header. The SG2010 supports Additional Frame Sizes from 0 to 8.

A write frame may contain byte masks in the first two quadwords of data. Two bits indicate whether the first and second quadwords are byte masked. If a quadword is byte masked, the eight byte masks for that quadword are placed in the least significant data byte of that quadword. If that least significant data byte contains valid data, then that valid data byte is placed in the least significant disabled byte position. (For more information about byte enables, see the *StarFabric Architecture Specification*.) If a PCI transaction has one or more bytes disabled subsequently to the first two quadwords, a new frame is generated to accommodate those byte-masked Dwords.

The orphan byte count (OBC) specifies how many empty bytes are in the frame, at the end of the data payload. The Orphan Byte Enable bit in the header indicates whether the OBC is present. When it is present, the OBC must be non-zero. The OBC is always located in the most significant byte of the first line (that is, bits [127:120]). Its value includes the byte containing the OBC, but does not include locations occupied by internal link overhead. Because the last data Dword is included as the fourth Dword in the first line of the frame, the OBC, when present, is located in the most significant byte of that piece of data.

The SG2010 does not use the orphan byte count to indicate unused bytes that are not Dword-aligned and in Dword granularities. In other words, the SG2010 uses the orphan byte count to indicate empty Dword locations only. The SG2010 uses byte masks to indicate disabled bytes and therefore is limited to including two quadwords of byte granular data.

Table 3–12 summarizes the data organization that the SG2010 uses when generating frames.

Table 3–12 Data Organization in Write Frames

Payload Size (Dwords)	Last Dword in Header?	Data in Last Line (Dwords)	Orphan Byte Count (bytes)	Link Overhead
1	Yes	1*	0	External
2, 6, 10, …	No	2	8	Internal
3, 7, 11, …	No	3	4	Internal
4, 8, 12, …	Yes	3	0	Internal
5, 9, 13, …	Yes	4	0	External

* This is also the first line; in other words, the data is located in the header line and this is a one-line frame.

3.4.2 Translating I/O and Configuration Write Transactions

PCI I/O and PCI configuration write transactions are translated into address-routed write frames. I/O and configuration operations are not supported in path-routed or multicast frames. Because configuration and I/O transactions are treated as delayed transactions, the SG2010 uses a frame Operation field of Write with Acknowledge (6h), where the acknowledge serves as a delayed write response in the fabric. If a parity error is detected on the received write data, then the Operation field is set to Write with Error with Acknowledge (7h).

The Target Region is set to either I/O (2h) or Configuration (3h). If a configuration transaction is translated, the SG2010 sets Address field bit [49] to 0 if it is a Type0 configuration transaction or to 1 if it is a Type1 configuration transaction.

Configuration and I/O writes are limited to a single Dword of data, so the SG2010 generates a single line frame. Because the header and data occupy the entire line, the link overhead is external. Because there is only a single Dword, the low four byte masks of quadword 0 (Qword 0) are used to cover all the data. The SG2010 sets the Orphan Byte Enable bit and the Byte Mask Qword 1 bits to 0.

Because only 32-bit addressing is supported, bits [49:32] of the Address field are set to 0 (with the exception of a Type1 configuration transaction, where bit 49 is a 1 and the remaining bits are 0).

All other fields are set in the same way as for address-routed write frames translated from a PCI memory write transaction.

3.4.3 Translating PCI Write Target Responses

The SG2010 generates a Write Acknowledge frame in response to a write frame with a "Write w/ Ack" operation.

When the SG2010 is the initiator of write transaction on the PCI bus and a response frame is required, it translates the target response into a Write Acknowledge frame. The SG2010 generates the values for the write acknowledge header fields as shown in Table 3–13.

Table 3–13 Write Acknowledge Header Fields

Frame Field	Value	Field Width	Bit Value
Class of Service	Provisioning (not I/O or Cfg) Address-routed (I/O or Cfg)	3	6h 4h
Operation	Write Acknowledge	3	1h
Path	Path transform of write frame path	–	–
Relaxed ordering	No	1	0b
Link Overhead	Internal	1	1b
Request Transaction Number	Transaction number obtained from write frame	16	–
Primary Event Code	Write Acknowledge	5	0h
Dwords Written	Number of Dwords transferred (0 – 32)	6	–
Request CoS	CoS of write frame	3	–
EMU Address	EMU 0	8	0
Address LSBs	PCI Address[15:2] of the of the first Dword	14	–

Because the Write Acknowledge frame carries no data, a single line frame with internal link overhead is generated.

The Failure Type header field contains error information, if any. In no errors were encountered during the delivery of the write, the SG2010 set the Failure Type to Fh (normal termination). Otherwise, the Failure Type field indicates the type of error encountered. Table 3–14 lists the possible Failure Types for a write transaction.

Table 3–14 Failure Types for Write Acknowledge

Failure Type	Description	Value
Range	Destination channel offset range failure	0h
Channel Lock	Destination channel path protection failure	1h
Channel Inactive	Invalid destination channel ID	2h
Parity Error	Target asserted PERR_L on PCI bus	3h
Target Abort	Target returned target abort on PCI bus	4h
Master Abort	No target response received on PCI bus	5h
Time Out	2^{26} clock cycle target response time-out on PCI bus	6h
Lockout	Cannot complete due to primary lockout condition (address routed only) or serial preload	7h
Hardware Routing Failure	Frame encountered a port down	9h
Software Routing Failure	Frame encountered an address routing failure, bad path, or multicast distribution failure within the fabric	Ah

Failure Types 0–2 are a result of checks performed on the incoming frame. When these failures occur, the SG2010 does not initiate the transaction on the PCI bus. It discards the transaction, returns the appropriate Failure Type in the Write Acknowledge frame, and signals an event. (For a description of destination channel operations, see Section 3.1.2.2.) Failure Types 9 and 10 are due to routing errors within the fabric. The remaining failure types are a result of transaction termination or parity errors on the PCI bus.

3.4.4 Translating PCI Read Transactions

A PCI read transaction has two components, the read request and the read completion (response). Both parts of the read transaction must be translated into a StarFabric frame to traverse the fabric.

3.4.4.1 Memory Read Requests

When the SG2010 is the target of a PCI memory read that is to be forwarded to the fabric, it translates the read request transaction into a StarFabric read request frame. PCI memory read transactions have several parameters that must be captured when translating the transaction into StarFabric format, as shown in Table 3–15.

Table 3–15 PCI Memory Read Request Translation Parameters

Parameter/Condition	Values	Comment
Bus Command	Memory Read	Read request operation used
	Memory Read Line	
	Memory Read Multiple	
Byte Enables	Not prefetchable	Byte-masked frame used

Table 3–15 PCI Memory Read Request Translation Parameters

	Prefetchable	Byte-enabled frame not used
Type of address hit	Bridge Base/Limit	Address-routed frame generated
	Gateway BAR2–BAR5	Path-routed frame generated

3.4.4.2 Address-Routed Read Request Header Generation

If an address-routed read request frame is generated, the corresponding frame field values listed in Table 3–16 are assigned.

Table 3–16 PCI Read Request Frame Field Values

Frame Field	Value	Field Width	Bit Value
Class of Service	Address routed	3	4h
Operation	Read Request	3	3h
Relaxed ordering	No	1	0b
Target Region	SAC (32-bit address) DAC (64-bit address)	2	00b 01b
Link Overhead	External	1	0b
Request Mode	Speculative	2	00b
Transaction Number	0 – 7, based on PCI Delayed Transaction queue entry	6	0h – 7h

3.4.4.2.1 Address Fields

At frame generation, the path of an address-routed frame is unknown and is initialized to 0s. The path is built dynamically as the frame travels through the fabric to the terminus. The path transform of this constructed path is for returning the read completion frame, or any path event frames, to the origin.

PCI address bits [49:2] form the Address field of the frame. If the address is a 32-bit address then bits [49:32] must be driven as 0. For a description of 64-bit PCI address support, see Section 3.1.4.7.6.

3.4.4.3 Path-Routed Read Request Frame Header Generation

The Segment Table, Path Table, and Source Channel Table properties define the values of many of the path-routed frame header fields. If a path-routed read request frame is generated, corresponding frame field values are assigned as shown in Table 3–17.

Table 3–17 Read Request Frame Field Values

Frame Field	Value	Field Width	Bit Value	Defined By…
Class of Service	Asynchronous Isochronous HP-Asynchronous HP-Isochronous Provisioning	3	0h 1h 2h 5h 6h	Segment Table
Operation	Read Request	3	0h	–
Request Mode	Speculative Prescriptive	2	00b 01b	Source Channel Table Source Channel Table
Relaxed ordering	No	1	0b	–
Link Overhead	External	1	0b	–
Transaction Number	0 – 7	6	0h – 7h	Based on PCI Delayed Transaction entry

3.4.4.3.1 Address-Related Fields

The address-related fields for path-routed frames are derived from the Segment and Path Tables, and source channel address translations as described in Section 3.1.2. The address-related fields are the 24-bit Path, the eight-bit Channel Number, and 42 bits of Offset, specifying bits [43:2].

3.4.4.4 Data-related Header Fields

A read request frame does not carry any data, but contains header fields specifying the type and amount of data to be retrieved. Both path-routed and address-routed frames use the same fields. Because a read request frame does not contain data, the Additional Frame Size is always 0.

The Byte Mask Enable field indicates whether byte masks are to be used during the read. The SG2010 only generates byte-masked frames for non-prefetchable reads as described in Section 3.3.2.1.

If byte masks are used, the SG2010 sets the Byte Mask Enable header bit to 1, and includes the byte masks in the eight-bit Byte Mask field. The Dwords Requested field is set to 1b to indicate that only a single Dword is to be read.

If the read is a prefetchable or prescriptive, then the Byte Mask Enable header bit is set to 0, and the number of Dwords requested is specified in the 10-bit Dwords Requested field. For a speculative read request frame, this number is derived from the cache line size and bus command as described in Section 3.3.2.2. For a prescriptive read request frame this number is obtained from the Source Channel Table.

3.4.5 Translating Configuration and I/O Read Requests

Both I/O and configuration PCI read request transactions are translated into address-routed read request frames.

The header fields used for I/O or configuration read requests are the same used for non-prefetchable PCI memory reads, with the exception that the Target Region is set to either I/O (2h) or Configuration (3h). If a configuration transaction is translated, the SG2010 sets address bit [49] to a 0 if it is a Type0 configuration transaction or to a 1 if it is a Type1 configuration transaction. Because only 32-bit addressing is supported, bits [49:32] of the address are set to 0 (with the exception of a Type1 configuration transaction, where bit 49 is a 1 and the remaining bits are 0).

The number of Dwords requested is always 1 and can support byte masks. If all bytes are enabled for that Dword, the SG2010 generates the PCI read request frame without byte masks (Byte Mask Enable = 0).

3.4.6 Translating PCI Read Completions

When the SG2010 is the initiator of a read transaction on the PCI bus, the Read Completion frame is used to return read data and any error information to the origin. All Read Completions use the same header format. The read completion header occupies three Dwords. The SG2010 generates values for the read completion header fields as shown in Table 3–18.

Table 3–18 Read Completion Header Fields

Frame Field	Value	Field Width	Bit Value
Class-of-Service	Same as read request CoS	3	–
Operation	Read Completion (no data parity error) Read Completion w/ Error (data parity error on PCI bus)	3	2h 3h
Path	Path transform of read request path	–	–
Relaxed ordering	No	1	0b
Request Transaction Number	Transaction number obtained from Read Request frame	16	–
Address LSBs	PCI address[15:2] of the first Dword of the first completion frame	14	–

The Failure Type header field contains error information. In no errors were encountered, the Failure Type is set to Fh (normal termination); Table 3–19 lists the otherwise possible Failure Types.

Table 3–19 Failure Types for Read Completion

Failure Type	Description	Value
Range	Destination channel offset range failure	0h
Channel Lock	Destination channel path protection failure	1h
Channel Inactive	Invalid destination channel ID	2h

Table 3–19 Failure Types for Read Completion

Target Abort	Target returned target abort on PCI bus	4h
Master Abort	No target response received on PCI bus	5h
Time Out	2^{26} clock cycle target response time-out on PCI bus	6h
Lockout	Cannot complete due to primary lockout condition (address routed only) or serial preload	7h
Hardware Routing Failure	Frame encountered a port down.	9h
Software Routing Failure	Frame encountered an address routing failure or a bad path within the fabric	Ah

Failure Types 0–2 are the result of checks performed on the incoming frame. When these failures occur, the SG2010 does not initiate the transaction on the PCI bus. It discards the transaction, returns the appropriate Failure Type in the Read Completion frame, and signals an event. (Destination channel operations are described in Section 3.1.2.2.) Failure Types 9 and 10 are due to routing errors within the fabric. The remaining failure types are a result of transaction termination or parity errors on the PCI bus.

3.4.6.1 Data Payload in Read Completion Frames

The data payload for read completion frames is organized the same way as for write frames. The orphan byte count and link overhead are also generated in the same way. For more information, see Section 3.4.1.3.

3.4.6.1.1 Data-related Header Fields

The remaining header fields are dependent on the amount and type of data to be sent.

The four-bit Additional Frame Size indicates how many lines of data follow the header. The SG2010 supports Additional Frame Sizes from 0 to 8.

A read completion may consume multiple frames. If the read completion frame is not the last frame, then the SG2010 assigns the Decomposition Type to Continuing (0b). Otherwise, the Decomposition Type is set to End (1b). Each frame in the read completion sequence must have a Decomposition Sequence Number. The first frame of the completion is assigned a Decomposition Sequence Number of 0, and the number is incremented with each frame generated for that completion.

3.5 Frame to PCI Translation

This section discusses the translation of the following frames into PCI transactions:

- Address-routed write frames
- Path-routed/multicast write frames
- Write acknowledge frames
- Address-routed read request frames
- Path-routed read request frames

- Read completion frames

This section does not discuss the reception of special frames, provisioning frames, or event frames.

Frames received by the SG2010 must conform to the following:

- Nine or fewer lines
- Size-optimized, as described in Section 3.4.1.3
- Use internal link overhead and are size-optimized; that is, the Orphan Byte Count is less than 12.
- Use either the Orphan Byte Count or byte masks:
  - For unaligned boundaries at the end of a frame
  - Subject to the amount of data supported by byte masks

The SG2010 ignores the Relaxed Ordering bit.

3.5.1 Translating Write Frames

When the SG2010 detects a write frame that is to be forwarded to the PCI bus, it first must translate the frame into a PCI transaction.

3.5.1.1 Frame Translation to Memory Write Transactions

Path-routed, multicast, or address-routed write frames with a target region of SAC or DAC are translated to PCI Memory Write or Memory Write and Invalidate (MWI) transactions.

For a more information about the SG2010 requirements for generating Memory Write and Invalidate commands, see Section 3.3.1.2.1.

The SG2010 generates the PCI transaction address differently depending on whether the frame is a address-routed frame or path-routed/multicast. If the frame is address-routed, the address is used as it is received in the frame; however, the SG2010 replaces address bits [63:50] for 64-bit addresses since those bits are not present in the frame Address field. The end result is the same address that was received on the initiating PCI bus. If the frame is path-routed or multicast, a destination channel address translation is performed on the frame’s Offset to generate the PCI address. For more information about address translation, see Section 3.1.

3.5.1.1.1 Translating the Data Payload

When translating the data payload, the SG2010 examines the Additional Frame Size, Link Overhead, Orphan Byte Count, Byte Mask Qword 0 and Byte Mask Qword 1 to determine where the valid data is and how it is arranged. If the Orphan Byte Count is present, it is located in the most significant byte of the first line. The payload organization is derived as shown in Table 3–20. The first Dword of the data payload for any frame of two or more lines is always located in the least significant Dword of the second line.

Table 3–20 Payload Organizations Supported in a Write Frame

AFS*	Internal Link Overhead	Orphan Bytes	Data Payload Organization
0	Must be 0	0 to 3	Single Dword write Data is fourth Dword in header line
> 1	0	0 to 3	Number of Dwords = (AFS × 4) + 1 (5, 9, 13, …) Last Dword is fourth Dword in header line
		4 to 7	Number of Dwords = AFS × 4 (4, 8, 12, …) Last Dword is fourth Dword in last line
		8 to 11	Number of Dwords = (AFS × 4) – 1 (3, 7, 11, …) Last Dword is third Dword in last line
		12 to 15	Number of Dwords = (AFS × 4) – 2 (2, 6, 10, …) Last Dword is second Dword in last line
	1	0 to 3	Number of Dwords = AFS × 4 (4, 8, 12, …) Last Dword is fourth Dword in header line
		4 to 7	Number of Dwords = (AFS × 4) – 1 (5, 9, 11, …) Last Dword is third Dword in last line
		8 to 11	Number of Dwords = (AFS × 4) – 2 (2, 6, 10, …) Last Dword is second Dword in last line
		12 to 15	Not supported – should be performed as an external link overhead frame with one less frame line

* AFS = Additional Frame Size

Once the amount and organization of data is determined, the byte masks are calculated. The existence of byte masks for the first two quadwords are indicated by the Byte Mask Qword 0 and Byte Mask Qword 1 header bits. If a bit is zero, that quadword has all bytes enabled. If a bit is a one, the eight byte mask bits are extracted from the least significant byte of the quadword. If the least significant byte position has valid data, the SG2010 copies that byte of data from the byte position of the least significant disabled byte in that quadword.

The PCI byte enables for the last Dword of data are calculated from the least significant two bits of the orphan byte count (OBC) as shown in Table 3–21.

Table 3–21 Byte Enable Calculation for Last Dword

OBC[1:0]	Last Dword BE_L	Description
00b	0000b	All bytes enabled
01b	1000b	Last byte disabled
10b	1100b	Last two bytes disabled
11b	1110b	Last three bytes disabled

Any data after the first two quadwords and before the last Dword are not covered by a byte mask mechanism. All bytes for those Dwords are enabled. If data is covered by both the Orphan Byte Count and the byte masks, the Orphan Byte Count covers the Dwords and the byte masks disable the bytes. In other words, the Orphan Byte Count must always be a multiple of four, and the PCI byte enables are based on only the byte mask bits.

If the operation is a Write with Error or a Write with Acknowledge with Error, the SG2010 forces a parity error for all the Dwords in the frame when it is driven on the PCI bus.

3.5.1.2 Recognizing DMA End-to-End CRC Write Frames

The SG2010 does not implement a DMA controller, and thus does not generate DMA read request or write frames. However, the SG2010 does recognize the write frame used for an end-to-end DMA check, although it does not actually track and perform the CRC operation. These single-line write frames are identified by the write with acknowledge operation (either with or without error) and a transaction number of 63. There is no class-of-service restriction. When the SG2010 detects this type of frame, it automatically generates a Write Acknowledge frame with a failure type of Normal. The write frame is dropped and not propagated to the PCI bus.

3.5.1.3 Frame Translation to Configuration and I/O Write Transactions

Address-routed write frames with configuration or I/O specified in the Target Region are translated to PCI configuration and I/O transactions, respectively. The StarFabric protocol limits the I/O and configuration write frames to single Dword, and therefore single line frames.

The transaction is translated and initiated in the same way as any address-routed write frame. The address bits [31:2] for the transaction are simply the frame Address field bits [31:2]. The SG2010 only uses the first four byte masks, if available, and the Orphan Byte Count to determine the byte enables of the transaction.

The SG2010 determines whether an incoming configuration write frame is a Type0 or Type1 frame by checking to see if Address [49] is a 0 or 1, respectively.

For more information about generating PCI I/O and configuration transactions, see Section 3.3.

3.5.2 Translating Write Acknowledge Frames

When the SG2010 receives a Write Acknowledge frame, it can be in response to one of the following:

- A configuration write, to generate a write response to the PCI initiator
- An I/O write, to generate a write response to the PCI initiator
- A path-routed or multicast write
  - Enabled by the source channel to request an acknowledge

  - Written to memory using an Event Message Unit (EMU)
- An event generated by another device

If the request transaction number is 0 – 7, then the write acknowledge is assumed to be in response to an outstanding I/O or configuration write transaction. The request transaction number selects the outstanding transaction buffer entry. If the transaction number is 62, the write acknowledge is in response to a software generated frame (see Section 3.11).

If the request transaction number is 61, the write acknowledge is an event frame or a frame in response to a path-routed or multicast write that was enabled to request a write acknowledge through a source channel. The write acknowledge is written to local memory by an Event Message Unit (EMU) Controller, as described in Section 3.7.3.3.1.

The SG2010 target response back to the PCI initiator for I/O and configuration writes is summarized in Section 3.3.7.

3.5.3 Translating Read Request Frames

When the SG2010 detects a read request frame that is forwarded to the PCI bus, it translates the frame into a PCI read transaction. A read request frame is always a single line frame using external link overhead since all four Dwords of the first line are used for the header.

3.5.3.1 Frame Translation to Memory Read Transactions

Path-routed or address-routed read request frames with a target region of SAC or DAC, are translated to PCI read transactions with a bus command of Memory Read, Memory Read Line, or Memory Read Multiple.

The SG2010 selects the appropriate bus command based on the number of Dwords requested and the cache line size. The guidelines the SG2010 uses for command selection are described in Section 3.3.2.3.

The SG2010 generates the transaction address differently depending on whether the frame is an address-routed frame or a path-routed frame. If the frame is address-routed, the address is used as it is received in the frame, however, the SG2010 replaces address bits [63:50] for 64-bit addresses. The end result is the same address that was received on the initiating PCI bus. If the frame is path-routed, a destination channel address translation is performed on the frame's offset to generate the PCI address. (Address translation is described in Section 3.1.)

If the Byte Mask Enable bit in the frame header is set, the SG2010 uses the eight-bit Byte Mask field for the byte enables of the transaction. The SG2010 reads only one or two Dwords in this case, depending on the value of the Dwords Requested field.

If the Byte Mask Enable bit in the frame header is not set, all bytes are enabled in the PCI read transaction. The number of Dwords requested ranges from 1 to 1K Dwords.

Note: *Crossing a 4KB boundary may result in unexpected behavior.*

If the Request Mode is speculative, the SG2010 initiates a single PCI read transaction and attempts to read up to the number of Dwords requested. If the Request Mode is prescriptive (either encodings 01b or 11b), then the SG2010 generates as many PCI read transactions as are necessary to read the number of Dwords requested. (For more information about the initiation of speculative and prescriptive read transactions, see Section 3.3.2.) If the Request Mode is Read-Modify-Write, it is a semaphore operation and the SG2010 does not initiate a transaction on the PCI bus. This is a CSR read operation with a semaphore side effect. Semaphores are described in Section 3.13.

The SG2010 stores the Transaction Number of the read request frame and uses it for the Request Transaction Number of the Read Completion frames.

The SG2010 ignores the Relaxed Ordering bit.

3.5.3.2 Frame Translation to Configuration and I/O Read Transactions

Address-routed read request frames with a Target Region of Configuration or I/O are translated to PCI read transactions with a bus command of configuration read or I/O read, respectively.

The transaction is translated and initiated in the same way as an address-routed memory read frame. The address bits [31:2] for the transaction are simply the Address field bits [31:2]. The SG2010 determines whether an incoming configuration write frame is a Type0 or Type1 frame by checking to see if Address [49] is a 0 or 1, respectively.

Although the SG2010 can only generate I/O and configuration read request frames requesting a single Dword, the SG2010 can read and support read request frames up to 1K Dwords, through either a prescriptive or speculative read.

Note: *Crossing a 4KB boundary may result in unexpected behavior.*

For more information about generating PCI I/O and configuration transactions, see Section 3.3.

3.5.4 Translating Read Completion Frames

When the SG2010 receives a Read Completion frame, it uses the Request Transaction Number to determine which outstanding delayed transaction the read completion data goes with.

A read completion to a PCI read transaction may consist of several frames. The Decomposition Sequence Number provides the ordering mechanism when there are multiple frames. The first frame of a read completion has a Decomposition Sequence Number of 0, and subsequent frames increment the value.

The Failure Type determines the target response back to the PCI initiator. The PCI target response is described in Section 3.3.7.

The SG2010 determines the data organization of the frame to extract the read data. The data organization is dictated by the same fields (Additional Field Size, Link Overhead, and Orphan Byte Count) as for a write frame, and organized in the same way (see Section 3.4.1.3). For read data buffering purposes, the SG2010 uses the Address LSBs field to determine whether the data is quadword aligned.

If the Operation is Read Completion with Error, then the SG2010 forces bad parity for all data delivered to the initiator. For more information about returning read data to a PCI initiator, see Section 3.3.2.5.

3.6 Ordering and Arbitration Rules

The SG2010 may have multiple transactions queued for initiation on the PCI bus, or multiple frames to be sent out a port. The SG2010 uses ordering rules and arbitration to determine which frame to send, or which transaction to initiate on the PCI bus.

3.6.1 PCI Transaction Ordering Rules

PCI transactions must follow a set of ordering rules as they are forwarded through a PCI hierarchy. The PCI ordering rules are summarized as follows:

- A PCI memory write can only be bypassed by a delayed write completion flowing in the same direction. No other transaction type flowing in the same direction can pass a PCI memory write.
- A PCI memory write must be allowed to bypass all other transaction types, except another PCI memory write.

All other transactions are not ordered with respect to each other and may be delivered in any order. Transactions flowing in opposite directions are not ordered with respect to each other and may be delivered in any order.

The SG2010 adheres to these ordering rules within a given class-of-service. When only address-routed frames through the Bridge function are supported, the class of service is address-routed and therefore all the above ordering rules are followed.

If the Gateway is enabled to support path-routed frames, the above rules apply within a class-of-service, but transactions converted to different classes-of-service are not ordered with respect to each other.

Additionally, frames received and forwarded to the PCI bus from different input ports are not ordered with respect to each other.

3.6.2 Frame Ordering Rules

Frames adhere to the following ordering rules:

- Frames in different classes-of-service are not ordered with respect to each other
- Frames with different output ports are not ordered with respect to each other
- For frames within the same class of service, and using the same output port:

- Write and read completion frames are delivered in order
- Read request frames are delivered in order
- Write and read completion frames may bypass read request frames

3.6.3 PCI Transaction Arbitration

The SG2010 may have several types of transactions queued to be initiated on the PCI bus. The SG2010 transaction arbiter determines the order in which these transactions are sent. Transactions first must meet the ordering rules outlined in Section 3.6.1. The SG2010 arbitrates among transaction types based on the CoS of its associated incoming frame. Additionally, transactions that have previously received target retry constitute another arbitration group. The arbitration groups are:

- Transactions from address-routed/asynchronous frames
- Transactions from isochronous/HP-isochronous frames
- Transactions from HP-asynchronous or multicast frames
- Transactions previously initiated that received target retry

Transactions in each basic transaction category are queued in a list in the order that they arrive from the link.

The transaction arbiter uses rotating priority where each arbitration group has a different number of entries in the arbiter. There are nine total entries in the transaction arbiter. Arbiter entries and their relative order of service are shown in Table 3–22.

Table 3–22 CoS Arbitration Entries

CoS Arbiter Entry	CoS
0	Retried transactions
1	Multicast/HP Asynchronous
2	Retried transactions
3	Multicast/HP Asynchronous
4	Isochronous/HP Isochronous
5	Retried transactions
6	Multicast/HP Asynchronous
7	Isochronous/HP Isochronous
8	Address-routed/Asynchronous

Retried transactions and multicast/HP-asynchronous transactions have three entries each in the arbiter, isochronous/HP-isochronous has two entries, and address-routed/asynchronous has one entry. The arbiter changes priority using the following rules:

- If no transactions are pending for arbitration, the priorities remain unchanged
- When transactions are pending for arbitration, the priority advances to the next highest numbered entry that has a pending transaction. The priority wraps from entry 8 to entry 0

For example, if the CoS arbiter previously selected entry 7 (Isochronous/HP Isochronous), and there are currently only multiple Multicast and Isochronous frames to send, the CoS arbiter selects entries 1 (Multicast), 3 (Multicast) and 4 (Isochronous/HP Isochronous) during the next three arbitration slots.

Once a transaction is completed, it is removed from the list. If a transaction is disconnected, but more data is to be transferred, the SG2010 moves that transaction to the tail of that list. If the SG2010 receives target retry in response to the transaction, then the transaction is placed on the tail of the Retry list. After data is transferred in response to a transaction that was on the Retry list, but the transaction is disconnected, then the SG2010 places the transaction at the tail of the original list.

3.6.4 Frame Arbitration

At any point in time, the SG2010 may have frames to send to one of its links from a variety of sources. These sources include:

- Frames converted from incoming PCI transactions
- Software-generated frames
- Read or write completion frames resulting from the SG2010 register accesses from the link
- High-threshold and low-threshold buffer credit bulk update special frames
- Event frames

The SG2010 frame arbiter determines the order in which these frames are placed in the output buffer for a link. The SG2010 uses the following fixed priority algorithm.

1.	Software generated frame	Incoming PCI transactions disconnected at next cache line boundary, subsequent transactions retried until frame sent.
2.	Event frames	–
3.	High threshold buffer credit bulk update Special frame	Incoming 64-bit PCI transactions disconnected at next cache line boundary. Incoming 32-bit transactions are not disconnected; the Special frame can be sent on a frame boundary.
4.	PCI transaction frame	Priority toggles between PCI frames and register completion frames, where the last type sent becomes the lower priority frame type between the two.
5.	Register access completion frame	
6.	Low threshold buffer credit bulk update Special frame	Only sent when there is nothing else to send.

3.7 Events

An event can result from an incoming interrupt or sideband signal (signal event), from a frame moving through the fabric (path event), or a chip event. A chip event can indicate that an error has occurred, or it can indicate a notification (informational event). A signal event is the assertion or deassertion of PCI interrupts INTA_L, INTB_L, INTC_L, INTD_L, or PCI sideband signals ENUM_L, PME_L and SERR_L.

Path events are generated when a path-routed frame moving through the fabric encounters a port condition or has a path that does not end at an edge node. A path event indicates that path invalidation should be performed. They are similar to chip events, but are always directed to the originator of the frame causing the event.

A routing event is a type of chip event that occurs when a frame encounters an error that is not a path event. A routing event is generated in a similar manner to a chip event, but the path event frame format is used, which includes the path of the frame that caused the error. Examples of routing events are path protection errors, and channel address range errors.

Events that occur in the fabric are processed in a sequence of stages. These stages may occur in the same device or in different devices. The stages are as follows:

- Event detection
- Event dispatch
- Event propagation
- Event handling

Event detection involves the detection and capture of the event in status bits at their source. Event dispatch is the process of determining where to send the event for handling, and if necessary, generating and sending an event frame. Event propagation is the routing of an event frame through the fabric. All StarFabric nodes perform event dispatch and propagation.

When an event is detected it is dispatched to an event handler which can implement several Event Message Units (EMUs). The event handler may be local (in the node where the event was detected), or remote (in another node attached to the fabric). If the event handler is remote then an event frame is generated to report the event to the event handler. The event frame is propagated through the fabric to the event handler.

When the SG2010 handles an event, it can assert a signal, write a message to memory, or both through an Event Message Unit.

The SG2010 supports event detection, dispatch, propagation, and handling.

3.7.1 Event Dispatch

In general, an event dispatcher translates an event into a path-routed frame with the write acknowledge operation (event frame). The SG2010 always uses Channel 255 when generating an event frame. This event frame is sent to a node that supports event handling.

The SG2010 detects and dispatches either signal events or chip events; because it is an edge node; the SG2010 does not dispatch path events.

Events are identified with a two-tiered coding system consisting of a Primary Event Code and a Secondary Event Code (described in Section 3.7.1.6). The five-bit Primary Event Code gives high-level information about the type of event, while the 14-bit Secondary Event Code provides finer level detail. Multiple events can be assigned to one Primary Event Code, and distinguished by their Secondary Event Codes.

The primary event codes are used as a lookup table indexes during event dispatch. Information obtained from the lookup tables specify where to send an event, and if necessary, to construct an event frame. The following Primary Event Codes do not have Chip Event Table entries in the SG2010:

0	Write acknowledge
1	Write message
2	Input signal assertions (uses Signal Event Table)
3	Input signal deassertions (uses Signal Event Table)
31	Path invalidation events
7, 12-26, 29-30	Reserved and not used by the SG2010

3.7.1.1 Event Tables

The SG2010 implements three tables that provide information for event frame generation and dispatch. These tables are the Chip Event Table (Section 4.6.8.1), the Signal Event Table (Section 4.6.8.2), and the Event Path Table (Section 4.6.8.3). When a chip event is detected, the Primary Event Code is used to index the Chip Event Table. When a signal event is detected the Signal Event Table is indexed based on the signal type and signal transition (assertion or deassertion). The Chip Event Table and Signal Event Table are formatted identically and contain the same fields, with the exception that local events cannot be generated using the Signal Event Table.

Each entry in the Signal Event Table and Chip Event Table contains the following information:

Destination Index	2 bits	An index into the four-entry Event Path Table
Send Mode	1 bit	When 0, selects polled mode for either sending event frames or handling them locally. When 1, selects list mode.
Local Destination	1 bit	If 1, indicates that the destination is local and an event frame is not dispatched; the local event handler is accessed. If a 0, the destination is remote and an event frame is conditionally dispatched. This field is reserved for signal events.
EMU Address	7 bits	Event Message Unit (EMU) address. EMU Address [6:1] selects the Event Message Unit at the destination. EMU Address [0] selects the operation to be performed at the EMU.

The Destination Index selects one entry in the Event Path Table. The Event Path Table entry specifies where the event is delivered. Each entry in the Event Path Table has the following information:

Path	21 bits	Path specification for the event frame
CoS	3 bits	The class-of-service of the event frame
Output Port	1 bit	The output port used for the event frame
Valid	1 bit	Valid path indication

If the Local Destination bit is 0, the SG2010 conditionally constructs and sends an event frame based on the information obtained from the Chip Event Table or Signal Event Table and the Event Path Table. If the Event Path Table entry is invalid, then a frame is not sent. The Primary Event Code, the EMU Address, and a Secondary Event Code are all part of the event frame header.

If the Local Destination bit is a 1, then an event frame is not constructed, but an access to a local Event Message Unit is performed. The EMU is selected by the EMU address [7:1] field in the Chip Event Table (same as for event frames), and the EMU operation by EMU address [0]. In this case, the path, CoS, and Output Port are not used.

The SG2010 sets the optional Event Data field to 0 in all chip and signal event frames that it dispatches.

For more information about generating chip event frames, see Section 3.7.1.3.

3.7.1.2 Signal Events

Signal events are any one of the following:

INTA_L asserts	INTA_L deasserts
INTB_L asserts	INTB_L deasserts
INTC_L asserts	INTC_L deasserts
INTD_L asserts	INTD_L deasserts
PME_L asserts	PME_L deasserts
ENUM_L asserts	ENUM_L deasserts
SERR_L asserts	

Because SERR_L asserts for only one cycle, SERR_L deassertion is not an event.

Signal events may be generated either with or without the Signal Event Table. Each signal may be individually enabled to use the Signal Event Table, or to use the default event information as described in Section 3.7.1.6.

Signal assertion and signal deassertion use Primary Event Codes 2 and 3 respectively. The Secondary Event Code identifies the signal that generated the event. Table 3–23 lists the codes and indexes into the Signal Event Table. If the Signal Event Table is not used to generate an event frame for the INT*x*_L signals, a different Secondary Event

Code is assigned to indicate to switches that a interrupt swizzle operation should be performed. This supports the staggered wired-OR of interrupt signals as described in the *PCI-to-PCI Bridge Architecture Specification, Rev 1.1*.

Table 3–23 Signal Event Codes and Table Indexes

Signal	Action	Secondary Event Code: Without Event Table	Secondary Event Code: With Event Table	Primary Event Code	Signal Event Table Index
INTA_L	Assertion	8	0	2	0
	Deassertion	8	0	3	1
INTB_L	Assertion	9	1	2	2
	Deassertion	9	1	3	3
INTC_L	Assertion	A	2	2	4
	Deassertion	A	2	3	5
INTD_L	Assertion	B	3	2	6
	Deassertion	B	3	3	7
PME_L	Assertion	4		2	8
	Deassertion	4		3	9
ENUM_L	Assertion	5		2	A
	Deassertion	5		3	B
SERR_L	Assertion	6		2	C

An event frame is typically sent for every occurrence of the events listed in Table 3–23. However, if a signal deasserts, and then asserts again before the deassertion event frame is sent, then the SG2010 may choose not to send a frame for either event. If the SG2010 sends an event frame for one of these cases, then it must send the event frame for both cases. This does not apply to the SERR_L signal. An event frame must be sent for SERR_L assertions, and event frames are never sent for SERR_L deassertion.

If the SG2010 is driving one of the event signals, then it blocks the sampling of the signals for signal event handling. That is, the SG2010 never generates a signal event frame as a result of signal assertion or deassertion by the SG2010.

3.7.1.2.1 Signal Event Dispatch without the Signal Event Table

Each signal can be individually enabled to use the Signal Event Table. The Signal Event Table is used only to dispatch a signal event frame if the corresponding Signal Event Table Enable bit is set by software. The Signal Event Table Enable bits are located in the Event Dispatch Control register in CSR space (described in Section 4.6.8.8). When the Signal Event Table is not used, signal events are assumed to be handled by a local processor when the SG2010 is a root or a Gateway-only leaf (Bridge disabled), and by a remote processor when the SG2010 is a leaf with the Bridge enabled.

By default, when the SG2010 is a root, or a leaf with the Bridge disabled, it ignores incoming interrupt signals (input signal pins). Signal events are only sampled and dispatched by the SG2010 when it is a leaf with the Bridge function enabled. When a leaf with the Bridge function enabled, the SG2010 uses the EMU address corresponding to the appropriate input signal and assertion/deassertion state, as shown in Table 3–24. Additionally, when the Signal Event Table is not used for signal event dispatch of an INT*x*_L signal, then the swizzle secondary event code is used instead of the standard secondary event code for that signal. The use of these event codes supports the staggered wire ORing of interrupt signals on add-in cards as described in the *PCI-to-PCI Bridge Architecture Specification, Rev 1.1*.

Table 3–24 Default EMU Address Assignments for Signal Events

PCI Signal	Default EMU Address	EMU (EMU Address [6:1])	Operation (EMU Address [0])
INTA_L assertion	02h	1	0
INTA_L deassertion	03h	1	1
INTB_L assertion	04h	2	0
INTB_L deassertion	05h	2	1
INTC_L assertion	06h	3	0
INTC_L deassertion	07h	3	1
INTD_L assertion	08h	4	0
INTD_L deassertion	09h	4	1
PME_L assertion	0Ah	5	0
PME_L deassertion	0Bh	5	1
ENUM_L assertion	0Ch	6	0
ENUM_L deassertion	0Dh	6	1
SERR_L assertion	0Eh	7	0

By default, if the SG2010 is a leaf with the Bridge function enabled, and it detects SERR# asserted, it only forwards a signal event frame if the SERR# Forward Enable bit is set in the Bridge Control register and the SERR# Enable bit is set in the Bridge's Command register. Otherwise, SERR# is ignored.

3.7.1.2.2 Signal Event Control Bit Setup

The SG2010 has three bits per interrupt signal that control the dispatch and handling of signal events:

- Input Signal Mask in the Event Dispatch Control register (described in Section 4.6.8.8) – enables or disables the sampling of the interrupt signal for event dispatch
- Signal Event Table Enable in the Event Dispatch Control register – enables or disables the use of the Signal Event Table for event dispatch
- Signal Output Disable in the Event Handler Control register (described in Section 4.6.8.9) – enables or disables the assertion of the interrupt signal when handling signal events

As a general rule, for a particular signal, the SG2010 should not be set up to sample signals and dispatch signal event frames while it is also being used to handle signal event frames and drive interrupt signals. If the Input Signal Mask is 0 (sampling is enabled), then signal event frames cannot be directed to the SG2010 event handler unless the corresponding Output Disable is set (to prevent the signal from being driven). Otherwise unpredictable behavior may occur.

Additionally, if the Signal Event Table is not enabled, then the SG2010 Input Signal Masks should never be set to 0 if the SG2010 is a root, or a leaf with the Bridge disabled.

Table 3–25 shows the SG2010 behavior in its different modes, for combinations of the signal event control bits.

Table 3–25 SG2010 Modes and Signal Event Control Bits

Root	Leaf without Bridge	Leaf with Bridge	Input Signal Mask	Signal Event Table Enable	Signal Output Disable
Illegal		Sampled and event frames sent to root; should not be a signal event target (Default: Leaf with Bridge)	0 (sampled)	0 (defaults used)	0
Illegal		Sampled and event frames sent to root; should not be a signal event target	0 (sampled)	0 (defaults used)	1
Sampled and event frames sent using signal event table; Should not be a signal event target.			0	1 (table used)	0 (drive signal)
Signals sampled; event frames sent using Signal Event Table			0	1	1 (disable signal)
Signals not sampled; event frames handled, signals driven locally (Default: Root and Leaf without Bridge)			1 (masked)	0	0
Signals not sampled; event frames handled, signals driven locally			1	1	0
Signals not sampled or driven			1	X	1

3.7.1.3 Chip Events

The SG2010 supports two modes of event dispatch – polled and list. These modes are specified on a per event basis in the Chip Event Table. In polled mode, and event frame is sent for every enabled (not masked) chip event, but only if the Event Status bit for that event bit has not been already set. The device must be read and the event bit cleared for a subsequent event of that type to generate a frame.

In list mode, an event frame is sent for every enabled chip event, regardless of the state of the Event Status bit. The status bits have no effect on the sending of event frames and reading the device or clearing the event bits is not required.

The process for generating and sending chip events is summarized in Figure 3–10 and the steps are described in the following sections.

3.7.1.3.1 Pending Events

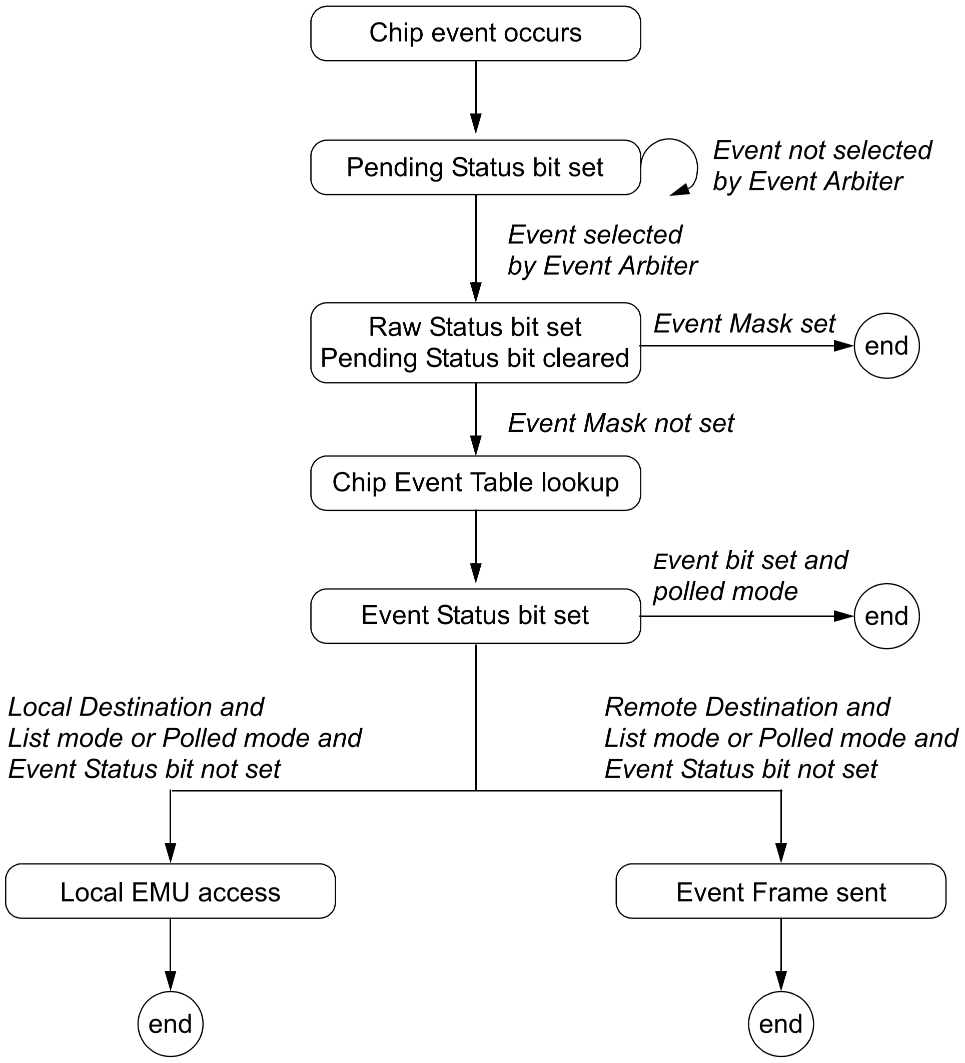
The SG2010 tracks pending event state for every chip event to determine which events have occurred but have not yet been dispatched. The SG2010 clears the bit when the event arbiter selects the event for dispatch. Since these bits are only needed by the SG2010 hardware, they are not visible to software through accessible register state. If an event occurs multiple times before it is dispatched, it is only dispatched once. Once an event is dispatched, a subsequent occurrence of that event causes it to be pending again, and dispatched after the event arbiter selects it.

When multiple events are pending, the event arbiter determines the order in which pending events are dispatched. A rotating algorithm is used such that all events are eventually serviced.

3.7.1.3.2 Event Mask

When the event arbiter selects a chip event for dispatch, the SG2010 checks the Event Mask register to determine whether to perform a Chip Event Table lookup. This register contains a bit for each chip event. The Event Mask has two register locations. One location is used to set a mask bit through a write-1-to-set (W1TS) operation; the other location is used to clear a mask bit through a write-1-to-clear (W1TC) operation. A read of either location returns the value of the mask register. When a mask bit is set for an event, a Chip Event Table lookup is not performed, a Event Status bit is not set, and an event frame is not sent.

Figure 3–10 Chip Event Dispatch Flow


Chip event occurs
Pending Status bit set
Event not selected by Event Arbiter
Event selected by Event Arbiter
Raw Status bit set Pending Status bit cleared
Event Mask set
end
Event Mask not set
Chip Event Table lookup
Event bit set and polled mode
Event Status bit set
end
Local Destination and List mode or Polled mode and Event Status bit not set
Remote Destination and List mode or Polled mode and Event Status bit not set
Local EMU access
Event Frame sent
end
end

3.7.1.3.3 Raw Event Status

When the event arbiter selects a chip event for dispatch, the SG2010 sets a corresponding bit in the Raw Event Status register regardless of the state of the Event Mask. The Raw Event Status register contains a bit for every chip event. A Raw Event Status bit is cleared by a software W1TC operation. The state of the bits in the Raw Event Status register have no affect on sending event frames or setting bits in the Event Status register.

3.7.1.3.4 Chip Event Table Lookup and Event Status Register

When the event arbiter selects a chip event for dispatch, and the corresponding mask bit is not set, a Chip Event Table lookup is performed. This lookup provides the Destination Table index used to select an entry in the Event Path Table. If the entry selected in the Event Table specifies list mode, then an event frame is sent, or, if a local event, the counter associated with Event Message Unit 0 is incremented. If the entry selected in the Event Table specifies polled mode, an event frame is sent or Event Message Unit 0 counter is incremented only if the Event Status bit is a 0 (before setting it due to this event).

The Event Status register contains a bit for every chip event. When a frame is sent or a Event Message Unit counter is accessed, the Event Status bit corresponding to that event is set.

3.7.1.3.5 Using Polled vs. List Mode

The list mode may be more desirable when the event handler, whether remote or local, writes information from each event frame to a list in its local memory, and the device is not polled. The polled mode may be desirable when the event causes an interrupt wire to assert at the event handler and the processor reads the SG2010 state to determine which events have occurred. Polled mode results in fewer event frames for remote event handlers, because setting the Event Status bit blocks any more frames for that event, and clearing the bit arms the dispatcher to send a frame for that event.

3.7.1.3.6 Routing Events

Routing events are a type of chip event caused by an address, channel, or path check on a frame. The frame format of a routing event uses the same format as the path event in that the path and input port of the frame that encountered the error are included in the event frame, and the width of the Secondary Event code is reduced to three bits.

When routing events are pending, the path, input port, and transaction number of each frame causing the event are saved. The SG2010 implements an eight-entry path cache for routing events. Any routing error type can have up to four entries in the cache. When a routing event is dispatched, the corresponding entry in the cache is freed. If a routing event should occur at the SG2010, and the path cache is full, or there are already four entries occupied by that type of routing event, then the SG2010 sets the Event Overrun status bit and does not generate a routing event for that case.

3.7.1.3.7 Chip Event Dispatch without the Chip Event Table (Default Mode)

After reset, the Chip Event Table is disabled and the SG2010 dispatches events in default mode. The Chip Event Table is used only to dispatch chip event frames if the Chip Event Table Enable bit is set by software. The Chip Event Table Enable bit is located in the Event Dispatch Control register in Gateway CSR space (described in Section 4.6.8.8).

In default mode, event dispatch and handling is local unless the SG2010 is a leaf with the Bridge function enabled. In the latter case, an event frame is dispatched with the root as the destination. In default mode, provisioning class-of-service is used for all event frames, and polled mode is used to dispatch them.

In default mode, chip events are directed to either EMU address 0h, 2h or Eh. EMU address 0h and 2h are intended for most event assertions (for example, INTA# in PCI), and specifies Operation 0 (typically a counter increment and signal assertion) at EMU 0 or 1. EMU address 0 is used for locally handled events (root or gateway-only leaf) and EMU address 2 is used for remotely handled events.

If the SERR# Enable in either the Bridge or the Gateway is set, certain events default to EMU address Eh. EMU address Eh is intended for system error event assertions (for example, SERR# in PCI) and specifies Operation 0 at EMU 7. If neither of the SERR# Enable bits are set, then these chip events are treated like all other chip events and directed to EMU address 0h for local events and 2h for remote events. Note that when the Chip Event Table is used to direct events to EMU address Eh, the SERR# Enable bits are not checked by the event dispatch logic. The system error events are:

- PCI Address Parity Error
- Master abort received on write without acknowledge
- Target abort received on write without acknowledge
- Data parity error received on write without acknowledge
- Target response timer expired
- Delayed transaction master time-out[1]

In order for an event to be dispatched, the corresponding Event Mask bit must be clear. The reset value of the Event Mask bits are all ones, and the reset value of both the Bridge and Gateway SERR# Enables are 0; therefore, by default, no events are sent unless these bits are programmed.

The default EMU for hot swap (INS and EXT) events is EMU 6, which is associated with ENUM_L assertion and deassertion (EMU addresses Ch and Dh respectively). These events are masked by standard hot swap registers, and are not handled by the Chip Event Table.

3.7.1.4 Path Events

A path event occurs when a path-routed frame encounters an error that may result in path invalidation. All these events have the Primary Event Code of 31. When a path event occurs, the event frame is sent to the origin of the frame. These errors are:

- Bad Path – a path-routed frame's path terminates in a switch (except for Channel 255 provisioning frames, which access a switch's register space)
- Port Down – a path-routed frame encounters a down port or a non-existent port on that device; no traffic can be sent or received using that port

Because the SG2010 is an edge node and either the origin or terminus of a frame, it does not generate path events. However, as an origin, the SG2010 can receive path event frames. Reception of these events and path invalidation is described in Section 3.7.4.

1. The Bridge function's Delayed Transaction Master Time-out bit has an accompanying SERR# Enable bit in the Bridge Control register specific to this event. This enable bit must also be set for this event to be reflected in the Raw Status register. This does not affect the master time-out for the Gateway function.

3.7.1.5 Hot Swap ENUM_L Events

The SG2010 implements a hot swap controller that has two events, INS (insertion) and EXT (removal) that can cause an ENUM_L signal to assert. Assertion of ENUM_L is controlled by a mask bit, EIM. The INS and EXT status bits, and the EIM mask bit are located in the Hot Swap Control configuration register (described in Section 4.7.7.3). Hot swap events are handled somewhat differently than chip events, as standard hot swap software polls and clears these bits in the Hot Swap Control register.

When either the INS or EXT bit is set or cleared and EIM is 0, the SG2010 determines whether to assert/deassert ENUM_L locally or remotely by checking the ENUM_L input signal mask in the Event Dispatch Control register (described in Section 4.6.8.8). If the signal is masked, then ENUM_L is asserted locally. If the signal is not masked, then an event frame is generated.

If ENUM_L is generated locally, then when EXT or INS are set with EIM clear, the ENUM_L Event Message Unit counter is incremented. When EXT or INS are cleared with EIM clear, the ENUM_L Event Message Unit counter is decremented.

If ENUM_L is generated remotely, then when EXT or INS are set with EIM clear, the Signal Event Table is indexed using the ENUM_L assertion index to generate an event frame. When EXT or INS are cleared with EIM clear, the Signal Event Table is indexed using the ENUM_L deassertion index to generate an event frame.

3.7.1.6 Event Codes

Primary Event Codes and Secondary Event Codes for the SG2010 events are shown in Table 27. Primary Event Codes 0 through 26, and code 31 are predefined, while Primary Event Codes 27 through 30 are device-specific assignments.

Table 3–26 Primary Event Code and Secondary Event Code Assignments

Event Type	Event	Primary Event Code	Secondary Event Code
Write Acknowledge	None	0	N/A
Write Message	None	1	N/A
Architected StarFabric Events			
Input Signal	INTA_L assertion	2	0
	INTB_L assertion	2	1
	INTC_L assertion	2	2
	INTD_L assertion	2	3
	PME_L assertion	2	4
	ENUM_L assertion	2	5
	SERR_L assertion	2	6
	INTA_L assertion w/ swizzle	2	8
	INTB_L assertion w/ swizzle	2	9
	INTC_L assertion w/ swizzle	2	A
	INTD_L assertion w/ swizzle	2	B

Table 3–26 Primary Event Code and Secondary Event Code Assignments *(Continued)*

Event Type	Event	Primary Event Code	Secondary Event Code
	INTA_L deassertion	3	0
	INTB_L deassertion	3	1
	INTC_L deassertion	3	2
	INTD_L deassertion	3	3
	PME_L deassertion	3	4
	ENUM_L deassertion	3	5
	INTA_L deassertion w/ swizzle	3	8
	INTB_L deassertion w/ swizzle	3	9
	INTC_L deassertion w/ swizzle	3	A
	INTD_L deassertion w/ swizzle	3	B
Link Events	Link x Down	4	[8:0] = 0 [13:9] = Link #
	Link x Fragile	4	[8:0] = 1 [13:9] = Link #
	Link x Up	4	[8:0] = 2 [13:9] = Link #
	Link x CRC Counter Wrap	4	[8:0] = 3 [13:9] = Link #
	Link x 8B/10B Counter Wrap	4	[8:0] = 4 [13:9] = Link #
	Link x Frame Count Wrap	4	[8:0] = 6 [13:9] = Link #
	Link x Line Count Wrap	4	[8:0] = 7 [13:9] = Link #
	Link x Empty Line Wrap	4	[8:0] = 8 [13:9] = Link #
Port Events	Port x Down	5	[9:0] = 0 [13:10] = Port #
Routing Events	Channel address range error	6	[2:0] = 0
	Channel path protection error	6	[2:0] = 1
	Invalid Channel ID	6	[2:0] = 2
	Address Routing Failure (non-Cfg)	6	[2:0] = 3
	SGF Done	8	0
Events	Event Overrun	9	0
System Errors	Address parity error	10	0
	Master abort on write w/o ack	10	1
	Target abort on write w/o ack	10	2
	Parity error on write w/o ack	10	3
	Target response timer expired on write w/o ack	10	4
	Response frame timer expired	10	5

Table 3–26 Primary Event Code and Secondary Event Code Assignments *(Continued)*

Event Type	Event	Primary Event Code	Secondary Event Code
	Delayed transaction master time-out	10	6
PCI Status	Detected Parity Error	11	0
	Signaled System Error	11	1
	Received Master Abort	11	2
	Received Target Abort	11	3
	Signaled Target Abort	11	4
	Master Data Parity Error	11	5
	Received System Error	11	6
	P2P Master Discard Timer	11	7
Reserved	Reserved	12 – 26	0
Device-Specific Events			
SG2010 Errors	Source channel address range error	27	[10:0] = 0 [13:11] = Ch#
	Invalid Segment Table Entry – frame discarded	27	1
	Response: No Matching Transaction Number	27	2
	Reserved	27	3
	Non-existent output port 1	27	4
	Parity Error on Provisioning Write	27	5
	Multicast: Distribution failure	27	6
	Address Routing Failure (PCI to fabric)	27	7
	Fabric Special Cycle (PCI to fabric)	27	8
SG2010 Notifications	Segment Table Entry Invalidated	28	0
Device-Specific Reserved	Reserved	29 – 30	0
Path Events			
Path Invalidation	Down port	31	0
	Reserved	31	1
	Bad path (turn count = 7)	31	2

3.7.2 Event Propagation

The class-of-service selected for an event frame can have ordering implications for the event frame as it travels through the fabric.

The ordering of event frames with respect to other frames (for example, write frames), can vary depending on the mechanism chosen and the class-of-service used. The following ordering rules should be considered when choosing the mechanism and CoS for event propagation.

- A path-routed event frame (generated by the SG2010) does not have the same ordering as an address-routed write frame.

- A path-routed event frame (generated by the SG2010) has the same ordering in the fabric as a path-routed write frame using the same CoS and path, although generation of event frames by the SG2010 is not ordered with the generation of other path routed frames.
- A message signaled interrupt (MSI[2]) write transaction (generated by a PCI device) that is translated into an address-routed write frame by the SG2010 Bridge function is ordered with an address-routed write frames.
- An MSI write transaction (generated by a PCI device) that is translated into a path-routed write frame by the SG2010 Gateway function is ordered with path-routed write frames using the same CoS.

When given a higher priority CoS, event frames have the ability to bypass lower priority CoS write and completion frames in the fabric.

In legacy PCI systems, interrupts by wire assertions bypass writes, and there are no ordering requirements between wire assertions and write data delivery. In order to most resemble this behavior, a high priority class-of-service, such as provisioning, should be used. When a PCI device uses MSI, interrupts in PCI systems are ordered with writes. In this case, the MSI transaction should be directed to the same the SG2010 function (Bridge or Gateway) that generated the write data frame and should use the same CoS if both are path-routed.

When event frames are received by the SG2010, any preceding writes of the same CoS to the PCI bus are completed before the SG2010 asserts an interrupt or delivers a message based on the event frame. EMU message writes also flush same CoS writes to PCI (See Section 3.7.3.2).

3.7.3 Handling Chip and Signal Event Frames

The SG2010 handles event frames by detecting the frame and conditionally asserting or deasserting an output signal (typically used as an interrupt), or writing event information to memory, or both.

The SG2010 recognizes event frames by detecting the Write Acknowledge operation. The Primary Event Code indicates whether it is a write acknowledge (Primary Event Code = 0), a write message (Primary Event Code = 1), or an event (all other Primary Event Codes). For a description of handling write acknowledges, see Section 3.7.3.3.1.

The event frame header contains a Event Message Unit (EMU) address that selects an Event Message Unit and an operation performed by the EMU in the SG2010. The SG2010 provides two types of event handling functions – event counters that control output signals (interrupts) and list controllers that manage writing event information to local memory. An EMU address used for the assertion of an event (operation=0) has both a counter increment operation and a Event Message Unit associated with it. An

2. Message signaled interrupt is a mechanism in PCI where an interrupt is performed through a memory write to a predetermined target rather than through a signal assertion.

Event Message Unit address used for the deassertion of an event (operation=1) is associated with a counter decrement operation. Event counters are described in Section 3.7.3.1, and Event Message Units are described in Section 3.7.3.2.

When an event frame is received, the functions associated with the specified EMU address are performed. Table 3–27 shows the association of EMU addresses with SG2010's event handling functions.

Table 3–27 SG2010 Functions Associated with EMU Addresses

EMU Address	EMU #	Operation	Functions	Notes
00h	0	0	EMU0 list controller EMU0 counter incr INTA_L control	Local event handling using INTA_L
01h		1	EMU0 counter decr INTA_L control	Local event handling using INTA_L
02h	1	0	EMU1 list controller EMU1 counter incr INTA_L control	Remote event handling using INTA_L
03h		1	EMU1 counter decr INTA_L control	Remote event handling using INTA_L
04h	2	0	EMU2 list controller EMU2 counter incr INTB_L control	–
05h		1	EMU2 counter decr INTB_L control	–
06h	3	0	EMU3 list controller EMU3 counter incr INTC_L control	–
07h		1	EMU3 counter decr INTC_L control	–
08h	4	0	EMU4 list controller EMU4 counter incr INTD_L control	–
09h		1	EMU4 counter decr INTD_L control	–
0Ah	5	0	EMU5 list controller EMU5 counter incr PME_L control	–
0Bh		1	EMU5 counter decr PME_L control	–
0Ch	6	0	EMU6 list controller EMU6 counter incr ENUM_L control	–
0Dh		1	EMU6 counter decr ENUM_L control	–

Table 3–27 SG2010 Functions Associated with EMU Addresses *(Continued)*

EMU Address	EMU #	Operation	Functions	Notes
0Eh	7	0	EMU7 list controller EMU7 counter incr SERR_L control	–
0Fh		1	EMU7 counter decr	–
10h	8	0	EMU8 list controller EMU8 counter incr	Not associated with a signal.
11h		1	EMU8 counter decr	Not associated with a signal.
12h – FFh			Reserved	Event Frame Dropped

3.7.3.1 Event Counters and Interrupt Signal Control

Event counters are used to control interrupt output signals based on the reception of event frames addressing the associated Event Message Unit. Each interrupt output signal (or sideband signal) is linked to an event counter as shown in Table 3–27. When the SG2010 receives an event frame with a Event Message Unit operation linked to a counter increment, the SG2010 increments the corresponding counter. When the SG2010 receives an event frame with an EMU operation linked to a counter decrement, the SG2010 decrements the corresponding counter. If the counter goes to zero, the number of deassertion event frames matches the number of assertion event frames. The counters stick at 0 when decremented and at 1FFFh when incremented; they do not wrap.

All interrupt output signals except for SERR_L function as level sensitive, shared interrupts. The SG2010 asserts an interrupt signal whenever any of the counters associated with the interrupt is non-zero. Note that INTA_L has two counters associated with it, one for local events and one for remote events. The interrupt remains asserted as long as any of the associated counters remain non-zero. Essentially, a non-zero state indicates an outstanding (unserviced) event. The SG2010 deasserts an interrupt when all of the associated counters are 0, meaning that all events have been serviced.

SERR_L is a special case in that it is asserted for a single cycle and requires no deassertion control. SERR_L assertion indicates a serious error condition and often causes a system crash to occur. The EMU7 counter is incremented when the EMU address linked to SERR_L assertion (EMU7, operation 0, or EMU Address Eh) is detected in an event frame. If enabled, SERR_L is asserted for one PCI clock cycle when the ENMU7 counter increments. In order to enable SERR_L for assertion, the SERR_L Output Mask bit must be clear in the Event Handler Control register, and the SERR_# Enable bit in either the Bridge or Gateway function must also be set.

The SG2010 provides software access to the event counters. For each counter, unique Dword addresses are provided for operations 0 and 1. Operation 0 is linked to the EMU counter increment and operation 1 is linked to the EMU counter decrement. If one of these dword locations is written through a register access with any value, the counter increments or decrements accordingly. Thus, a local processor can control the deassertion of an interrupt signal by decrementing the counter (writing the dword location associated with the counter decrement) until it reaches zero. If either Dword location of the counter register is read, the value of the counter is returned. Each counter is 13 bits wide, and sticks at 1FFFh when incrementing and 0 when decrementing.

Proper incrementing and decrementing of the counters is essential when propagating a change in state of a remote interrupt signal to a local interrupt signal. Chip events cause a counter increment, but never a counter decrement. Software is responsible for decrementing the event counter when chip events are serviced.

3.7.3.1.1 Handling PCI Legacy Signal Interrupts

An event frame that has a Primary Event Code of 2 or 3 (signal assertion or deassertion) and a Secondary Event Code of 8h through Bh (legacy PCI INTA# through INTD#) specifies that an interrupt swizzle operation must be performed as it is routed through the levels of PCI bridging in the fabric as described in the *PCI Bridge Architecture Specification, Revision 1.1*. This includes the terminus, which can be an SG2010 that performs event handling.

If the SG2010 detects an incoming signal event frame with the above event codes, it modifies the EMU Address as shown in Table 3–28.

Table 3–28 PCI INTx# EMU Address Swizzle Modification

Incoming EMU Address	EMU Address Used (only for Secondary Event Codes 8h - Bh)	
	Input Port (PCI Device #): 0	Input Port (PCI Device #): 1
0	0	
1	1	
2	2	4
3	3	5
4	4	6
5	5	7
6	6	8
7	7	9
8	8	A
9	9	B
A	A	2
B	B	3
All Others = X	X	

3.7.3.1.2 Signaling Interrupts through PCI MSI transactions

A PCI MSI transaction can be considered as the transaction equivalent of asserting an interrupt signal (INTA_L, INTB_L, INTC_L, INTD_L). However, instead of asserting a wire, the SG2010 initiates a single-Dword PCI memory write transaction using a programmed address and data payload. If the PCI Message Signaled Interrupt (MSI) unit is enabled and that particular counter is also enabled to use PCI MSI, an MSI write transaction is generated instead of asserting a signal.

The PCI MSI unit is enabled on two levels. The standard configuration register set for PCI MSI has an enable bit for the unit. Additionally, the SG2010 implements MSI enables for each counter that controls an INT*x*_L wire. Thus, PCI MSI can be enabled for only local events through EMU0, or selectively for remote events using the other EMUs associated with INTx_L pins.

If both the PCI MSI message unit is enabled and the associated EMU is also enabled, then both a PCI MSI message and an EMU message are generated when the corresponding EMU address is specified in an event frame. For a description of the Event Message Unit, see Section 3.7.3.2.

The SG2010 requests four MSI messages from the host, one for each interrupt signal output. If all four messages are granted, the MSI data payload then consists of the data prefix assigned in the Message Data register, with the two low data bits equal to the INTx_L encoding of the asserted interrupt. If the host assigns less than four messages, the SG2010 always uses the same data payload assigned by the Message Data register.

3.7.3.2 Event Message Buffer Control

In addition to controlling EMU counters, Event Message Units (EMUs) enable event information to be written to a location in memory.

The SG2010 implements nine Event Message Units. Each Event Message Unit manages the writing of event data to an event message buffer, one buffer per EMU. The actual event message buffers reside in local memory. Event data is written to the end, or tail, of the buffer. The processor retrieves this event data from the beginning, or head, of the buffer. SG2010's control logic implements the functionality required to write event data to the tail of the event message buffer. Software maintains the pointer for the head of the event message buffer.

The size of each event message buffer is individually programmable to one of four possible values: 0 (disabled), 4KB, 16KB, or 64KB. The tail pointer is a quadword-aligned PCI address that consists of an upper 32-bit base address, which locates the list in 64-bit memory, a lower 32-bit base address, which locates the list in 32-bit memory, and the Event Message Unit tail pointer that selects the address location of the tail within the buffer. The width of the overall (upper plus lower) base address and EMU tail pointer depend on the size of the buffer corresponding to that EMU. Because the address is quadword aligned, the low three address bits are always zero. A counter is used to manage the remaining tail pointer bits. The counter is 14 bits wide. Because only 13 tail pointer bits are needed for the largest buffer, the most significant counter bit(s) indicates whether the counter has wrapped. For the smaller buffers, since there are multiple extra counter bits, it also shows to a limited extent how many times it has wrapped. If the 14-bit value of the counter reaches all 1s, it wraps to 0 on the next EMU operation.

Table 3–29 shows the base address and tail pointer widths for each buffer size, as well as the counter bits used for the tail pointer bits, and the extra counter bits that indicate that the counter has wrapped.

Table 3–29 Event Message PCI Address Components

Buffer Size	Base Address*	Tail Pointer	Counter	Wrap Count
0	N/A	N/A	N/A	
4KB	Addr[63:12]	Addr[11:0]	Ctr[8:0]	Ctr[13:9]
16KB	Addr[63:14]	Addr[13:0]	Ctr[10:0]	Ctr[13:11]
64KB	Addr[63:16]	Addr[15:0]	Ctr[12:0]	Ctr[13]

* Base address consists of Upper Base Address [63:32] and Lower Base Address [31:*x*]

All Event Message Units share the same value for upper base address bits [63:32] for their event message buffers. Each Event Message Unit has an individual lower base address register for the remaining base address bits [31:*x*]. Each buffer must be located on a size-aligned address boundary; for example, a 64KB buffer must be located on a 64KB-aligned address boundary.

3.7.3.3 Event Message Data Formats

When the SG2010 receives an event frame that has an EMU address associated with an event message buffer, it initiates a PCI memory write transaction using the current event message PCI address. The PCI transaction data is two quadword-aligned Dwords as shown in Table 3–30.

Table 3–30 Event Message Data Payload

Byte 3	Byte 2	Byte 1	Byte 0		
Event Frame Path Specification			Input Port [2:0]	Primary Event Code	**Dword 0**
Dword 2 of the event frame					**Dword 1**

The event frame path specification includes the turn count of the path as it was received, and the path transform (invert and reverse) of the event frame path (that is, the path from the SG2010 to the dispatcher of the event). If the event was dispatched locally, the input port and turn count are both set to 0, and all turns in the path are set to 7 (octal).

After the write transaction is completed, the SG2010 increments the tail pointer to point to the next quadword-aligned address. If a PCI error is encountered when writing the event message transaction, then the SG2010 disables the Event Message Unit that generated the transaction.

3.7.3.3.1 Using EMU 0 for Memory Write Acknowledges

The SG2010 uses EMU 0 to handle write acknowledge frames received in response to either path-routed or multicast writes.[3] When it receives a write acknowledge frame that is not destined for either the I/O or configuration delayed transaction logic, or for the SGF function[4], then the SG2010 directs the write acknowledge frame to EMU0. The

SG2010 increments event counter associated with EMU0, conditionally asserting INTA_L. The payload for the event massage for write acknowledges is the same as for other event frames and is shown in Table 3–31. The write acknowledge path specification includes the turn count of the path as it was received, and the path transform (invert and reverse) of the event frame path (that is, the path from the SG2010 to the node that generated the write acknowledge).

Table 3–31 Write Acknowledge Event Message Payload

Byte 3	**Byte 2**	**Byte 1**	**Byte 0**		
Write Acknowledge Path Specification			Input Port [2:0]	Primary Event Code (0)	**Dword 0**
Dword 2 of the Write Acknowledge frame					**Dword 1**

3.7.3.3.2 Using an Event Message Unit for Messaging

Event Message Units can be used for delivering write messages to local memory. Each Event Message Unit has two CSR Dword locations associated with it. In order to trigger a message write, software writes the lower (even) dword associated with operation 0 of that Event Message Unit. The write must be a single line frame (one Dword data payload) to a Channel 255 offset or to an address-routed memory address. Multiple line frames have no effect – no message is written and the counters are not affected. Additionally, write access to these register locations through I/O space have no effect. When the EMU lower dword register is written, the data payload is written to the memory location addressed by the current tail pointer value for that Event Message Unit. That is, the SG2010 manages the EMU as if an event frame addressed it. Additionally, the associated event counter is incremented, and the interrupt signal is conditionally signaled through an interrupt wire or MSI write.

When the SG2010 performs an event message write to memory based on a CSR access, it uses the Primary Event Code for write message (Code 01h). The payload used for the write to memory is still two Dwords, but the second Dword is the data payload of the write frame (third Dword of the frame instead of second Dword) as shown in Table 3–32. The write frame path specification includes the turn count of the path as it was received, and the path transform (invert and reverse) of the event frame path (that is, the path from the SG2010 to the origin of the write frame).

3. The SG2010 does not generate address-routed memory write with acknowledge frames, except through the SGF function. In this case, the acknowledge would be handled by the SGF function.

4. Which is handled separately

If the CSR access is local (from PCI), the counter is incremented (and the corresponding interrupt conditionally asserts) but an event message is not created, for both I/O and memory accesses.

Table 3–32 Write Message Payload

Byte 3	Byte 2	Byte 1	Byte 0		
Write Frame Path Specification			Input Port [2:0]	Primary Event Code (1)	**Dword 0**
Dword 3 (data payload) of the Write frame					**Dword 1**

3.7.4 Handling Path Event Frames

Switches generate path events when a frame has a bad path specification (erroneously ends at a switch due to a turn count of 7), or a down port is encountered. A path event is always returned to the origin of the frame. A Primary Event Code of 31 is used for path events, and the Secondary Event Code identifies the type of path event.

When the SG2010 receives a path event frame, it both directs it to the Event Message Unit specified by the EMU address in the frame and performs path invalidation.

3.7.4.1 Path Invalidation Sequence

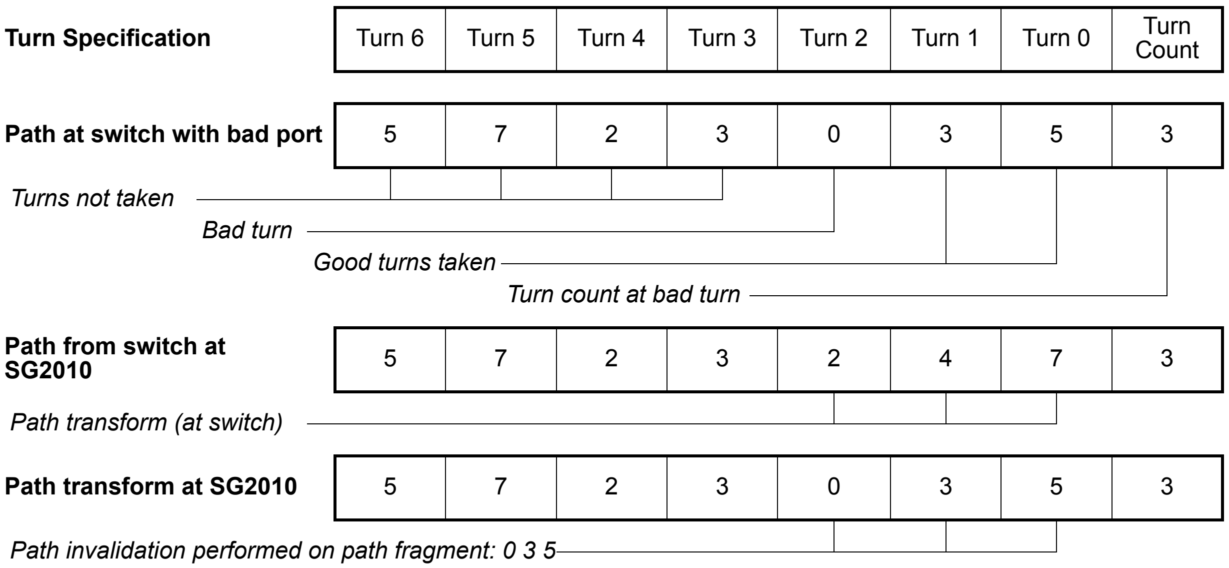
When path invalidation is performed, each entry in the Segment Table and the Event Path Table is checked and marked invalid if it is part of the invalid path.

When performing path invalidation, the SG2010 returns target retry to all PCI transactions translating to path-routed frames. This prevents any more transactions from being accepted that could possibly use the same bad path. This retry condition persists until the path invalidation sequence is completed. Accesses to the SG2010 register space and transactions translating to address-routed frames may still be accepted.

3.7.4.1.1 Calculating Bad Path Prefix

The bad path prefix is obtained from the path field in the event frame. The SG2010 calculates the bad path prefix, or the number and direction of turns to the port in question. The turn count in the received frame indicates the number of active turns in the path, up to and including the bad turn. For example, assume the path specification shown in Figure 9. A turn count of 3 means that the first two turn fields are valid, and that the event occurred on the third turn. From this information, the SG2010 creates the path prefix by performing a path transform on the active turns by reversing their order, and inverting their bits. In the example below, when the path event arrives at the SG2010 the number of active turns is 3, and turn0 = 7, turn1 = 4, and turn2 = 2. The bad turn is turn2. The SG2010 inverts and reverses these three turns to create the path prefix turn0 = 5, turn1 = 3, and turn2 = 0. Path invalidation is performed on this prefix.

Figure 3–11 Path Invalidation Prefix Operation



3.7.4.1.2 Invalidating Segment Table Entries

After the SG2010 has the bad path prefix, it starts the invalidation process for the Segment Table. The SG2010 steps through every Segment Table entry and performs the following operations:

- Obtain the corresponding path from the path table using the path index field.
- If the path has the same prefix as the bad path, and the output port matches the port on which the path event frame was received, then invalidate the segment table entry by setting bit 3 of the four-bit path length field to 1.
  - If it does not match, move to the next segment table entry
  - If the segment table is configured to be redundant, both the primary and secondary paths in the segment are checked

If one or more Segment Table entries are marked as invalid, the SG2010 sets the Segment Table Entry Invalidated event bit.

3.7.4.1.3 Invalidating Event Table Entries

Path invalidation is performed on the Event Path Table as well. The SG2010 compares the bad path prefix against the same path bits in all four entries of the Event Path Table. For every match, the SG2010 sets the valid bit of that Event Path Table entry to 0.

3.7.4.1.4 Handling Multiple Path Invalidation Events

The SG2010 implements a single entry cache that holds the last path invalidated, as well as a software-accessible Disable Last Path Invalidation bit. The SG2010 compares the path of received path events to the path in the cache. If the paths match and the Disable Last Path Invalidation bit is set, then the SG2010 does not perform path invalidation and discards the path event frame. If the paths do not match, then the SG2010 performs a path invalidation on the received path.

The SG2010 sets the Disable Last Path Invalidation bit when it loads the cache after performing path invalidation. The cache is loaded with the transformed bad path, including turn count. Software clears the Disable Last Path Invalidation bit to enable that path for path invalidation on subsequent path events. The default state of the valid bit after reset is 0.

3.7.4.2 Path Invalidation Modes

The SG2010 does not support hardware synchronization or ordering between frames using a path that becomes invalid and subsequent frames that use a redundant route.

Depending on the type of traffic the SG2010 is forwarding, it may be desirable to support redundant paths for some segments but not others. For low latency traffic where data integrity and ordering is less important, redundant paths may be desired. For control traffic where data integrity and ordering is more important than latency, the SG2010 can simply invalidate the path and allow software to handle Segment Table and Path Table updates to ensure data ordering and recovery (data can be lost when a port goes down).

Each Segment Table entry may or may not support a redundant path. If any entries are to support redundant paths, the Redundant Segment Table bit must be set for the corresponding BAR in the Gateway Chip Control Status 0 register in CSR space. This configures each segment for that BAR to contain two valid entries. If a redundant path is not desired for that segment, then software can mark the secondary Segment Table entry invalid. If a redundant path is desired, software marks both the primary and secondary entries valid.

3.7.4.3 Retiring Transaction Numbers

When the SG2010 receives a path event, the header contains the transaction number and operation of the original frame sent by the SG2010 that caused the event. If the transaction number and operation matches one stored in an PCI Delayed Transaction buffer entry, that transaction number may be immediately retired and the buffer entry is available for another transaction.

3.8 Reset and Initialization

The SG2010 initialization process consists of the following components:

- Reset
- Link synchronization
- Reset propagation
- Serial preload
- Fabric Enumeration

  - Component identification
  - Fabric ID assignment
  - Link to port mapping
- Line credit initialization

The SG2010 implements several reset mechanisms. Not all initialization components are a part of every reset mechanism. Each reset mechanism, and the initialization flow that accompanies it, is discussed in the following sections.

3.8.1 Reset

The SG2010 supports the following reset mechanisms:

- PCI RST_L input signal – power-up platform reset
- LRST_L input signal – power-up local reset; also used for hot swap
- Fabric Reset bit in Fabric Reset register – resets the entire fabric
- Node Reset bit in SFC Control register – resets the chip
  - If Propagate Maskable Reset bit in the same register is also written with a 1, a maskable reset comma is sent out all links prior to the chip reset
- Secondary Reset bit in Bridge configuration registers – resets PCI registers of downstream nodes and PCI devices
- Power Management state transition from $D3_{hot}$ to D0 – power management reset which clears PCI registers
- Maskable comma character – propagated reset received from the StarFabric interface
- Unmaskable comma character – propagated reset received from the StarFabric interface

SG2010 reset mechanisms are summarized in Table 33.

Table 3–33 SG2010's Reset Mechanisms

Mechanism		Link Synch. and Credit Init.	Propagation	Serial Preload	Fabric Enumeration
H/W	RST_L	Yes	Maskable Reset (Root)	Yes	Root
	LRST_L	Yes	Maskable Reset (Root)	Yes	Root
	Maskable Reset	Yes	Maskable Reset	Yes	Root
	Unmaskable Reset	Yes	Unmaskable Reset	Yes	Root
	Address Routed Reset	No	Address Routed Reset (if Bridge enabled)	No	No
S/W	Fabric Reset	Yes	Unmaskable Reset	Yes	Root
	Node Reset	Yes	If Propagate Maskable Reset bit written with 1	Yes	No*
	Secondary Reset	No	Address Routed Reset	No	No
	PM D3 $\rightarrow$ D0	No	Address Routed Reset	No	No

* Only the sending of You Are frames is inhibited during this case. I Am and Set Credit frames are still sent.

The above reset mechanisms, except for Secondary Reset, Power Management (PM) reset, and the address routed reset comma, clear all SG2010 state and data buffers.

RST_L and LRST_L assertions propagate a maskable reset comma if the SG2010 is the root, otherwise these signals are considered local resets and are not propagated into the fabric. RST_L and LRST_L are identical in function, with the exception that a RST_L assertion resets the clock and data recovery circuitry of the links, but and LRST_L assertion does not.

The Secondary Reset, located in the Bridge Control configuration register reset standard PCI registers in downstream nodes and resets downstream PCI devices. Because the SG2010 Port Map Tables contain address decoding data for downstream nodes, when the Secondary Reset bit is set, the SG2010 clears all the Port Map Table registers. The Port Map Table Enable and Smart Address Enable bits are not reset.

The Power Management reset occurs when software changes the power management state from the D3 low power state to the D0 high power state. When this power management state transition occurs, the SG2010 clears standard PCI registers with R/W and R/W1TC access. Standard PCI registers are those located within configuration offsets 00h and 3Fh, as well as the enhanced capability functions MSI and Power Management. If the Bridge is enabled, the SG2010 transmits an address routed comma out both links.

If an address routed comma is received by the SG2010, and the Reset Disable bit in the Port State Control and Status register for that port is clear, the standard PCI registers with R/W and R/W1TC access are reset to their default values and an address routed reset is propagated out all ports. If the Reset Disable bit is set, the address routed reset comma is ignored.

A node reset propagates a maskable reset comma if the Propagate Maskable Reset bit in the SFC Control register is written with a 1 with the same access. Reset propagation is described in Section 3.8.1.1. If a reset does not propagate, only the SG2010 is reset. All state and data buffers are cleared. SG2010 link partners participate in link resynchronization and line credit exchange as a part of the SG2010 initialization flow.

The SG2010 performs serial preload, link synchronization, component identification, and line credit initialization after every type of reset except for Secondary Reset, Power Management reset, and address routed reset.

Although all nodes identify themselves after link synchronization by sending the I Am frame, only the root can initiate fabric enumeration by sending the You Are frame.[5] All other nodes perform fabric enumeration only when a link partner transmits a You Are enumeration frame to them. When the SG2010 is the root, it initiates fabric enumeration after any propagating reset. If reset is not propagating, or the SG2010 is a leaf, it does not initiate fabric enumeration. Fabric enumeration is described in Section 3.8.4.

Figure 3–12 is a summary of the initialization flow. Subsequent sections describe each step.

3.8.1.1 Reset Propagation

Reset propagation in the fabric relies on the transmission and reception of comma characters (well-defined 10-bit encodings) between link partners. Links must be synchronized to propagate a reset, but the Traffic Enable bit does not need to be set. StarFabric protocol defines three comma characters for reset propagation: a maskable reset comma character, an unmaskable reset comma character, and an address routed reset comma character. Unmaskable resets received by a node unconditionally reset that node. Maskable comma characters reset the node only if the Reset Disable bit in the Port State Control and Status register corresponding to the port where the comma was received is off (0), otherwise the maskable reset is ignored (and not propagated).

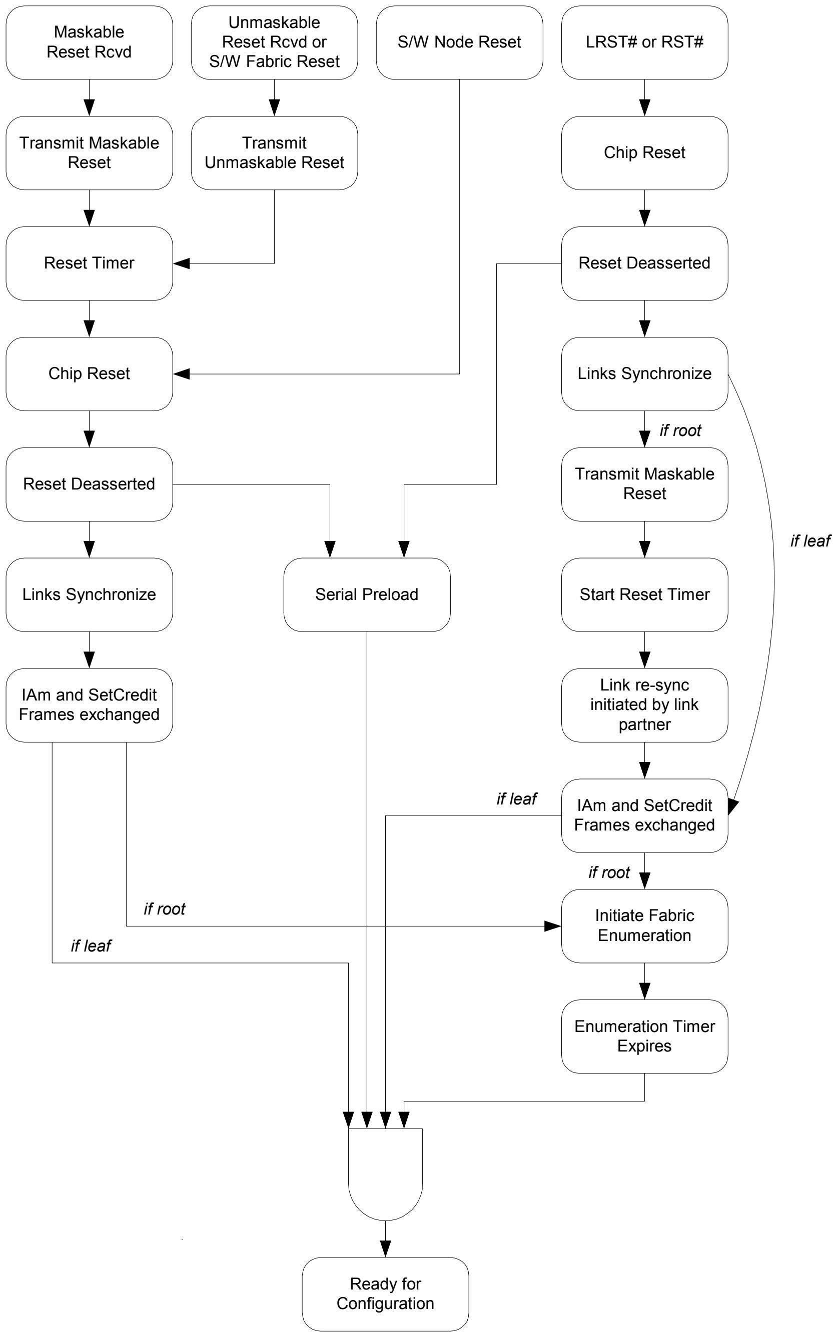
Address routed reset commas are also masked with the Reset Disable bit. If the bit is clear, the SG2010 propagates the address routed reset comma out all links when one is received.

3.8.1.1.1 Sending Propagating Resets

To propagate a maskable reset comma, a unmaskable reset comma, or an address routed reset comma, the SG2010 sends four of the comma characters in succession out each of its differential pairs for all links. When this transmission occurs, the SG2010 starts a Reset Mask Timer. The Reset Mask Timer prevents infinite reset loops caused by reset propagation by filtering all received reset comma characters. Until the Reset Mask Timer expires after 128-78MHz clock cycles for a maskable or unmaskable reset or 1024-78MHz clock cycles for an address routed reset, the SG2010 ignores all subsequent reset comma characters that it receives

5. Unless it is generated using a software generated frame function.

Figure 3–12 SG2010 Initialization Flow



3.8.1.1.2 Receiving Propagating Resets

To detect an incoming propagating reset on a link, the SG2010 must detect four reset comma characters on at least one of its operational differential pairs. Otherwise, the comma characters are ignored.

When the SG2010 detects an unmaskable reset, the SG2010 transmits the unmaskable reset out all of its links and starts its Reset Mask Timer as described in the previous section. The SG2010 then performs the rest of initialization flow after the reset timer expires and a chip reset is performed.

When the SG2010 detects a maskable reset on a link, it first checks the Reset Disable bit in the corresponding Port State Control and Status register. If the Reset Disable bit is 1, the SG2010 ignores the maskable reset and does not propagate it. If the Reset Disable bit is 0, the SG2010 sends a maskable reset out all of its links and starts its Reset Mask Timer as described in the previous section. The SG2010 then performs the rest of the initialization flow.

The Reset Disable bit is reset to 0, which allows any link partner to reset the SG2010. Fabric enumeration sets all Reset Disable bits except for the ones corresponding to the root port. This allows a propagating reset to flow down the tree hierarchy from the root port but not from any other direction. If the SG2010 is a root, both ports have their Reset Disable bits set by fabric enumeration. If the SG2010 is a leaf, only a non-root port has its Reset Disable bit set.

3.8.1.1.3 Blocking Maskable Resets

The SG2010 provides an implementation-specific mechanism to block incoming maskable resets, and to manage the links when such a reset is blocked. This mechanism prevents a reset of the fabric from affecting an operating subsystem.

The Maskable Reset Mode bit in the Chip Control register in Gateway Configuration space enables this mechanism. When the Maskable Reset Mode bit is 0, incoming maskable resets are conditioned on the Reset Disable bit as described in Section 3.8.1.1.2. When the Maskable Reset Mode bit is set, an incoming maskable reset comma does not reset the chip, regardless of the state of the Reset Disable bit. However, the SG2010 will take one of the following actions:

- If the SG2010’s Fabric ID is 7/7/7777777 (leaf reset value), the SG2010 sets the Traffic Enable bit in the Link State Table. It is assumed that the credit counters and the sequence numbers between the link partners are consistent.
- If the SG2010’s Fabric ID is not 7/7/7777777 (enumeration has occurred), the SG2010 clears the Traffic Enable bit in the Link State Table, and then subsequently resets the Fabric ID. Software must intervene to bring the link up (as described in Section 3.9.1.1) and the SG2010 may be re-enumerated.

3.8.1.2 RSTO_L Reset Output Signal

The SG2010 implements a reset output signal, RSTO_L, that may be used to reset devices on the PCI bus. This signal is asserted for all reset mechanisms, for root and leaf, excepting when the Propagated Maskable Reset is written with a 1 without setting the Node Reset bit. When asserted, the SG2010 maintains the assertion for the minimum PCI specification time of 100μsec, or in the case where the Secondary Reset bit was set, when software clears the Secondary Reset bit.

3.8.2 Serial ROM Preload

The SG2010 optionally performs a serial preload operation to write SG2010's register state after reset. All registers with write access, and selected read-only registers, can be written through the serial preload operation. For more information about preload register access, see Section 4.3

Serial preload runs in parallel with link synchronization, component identification, fabric enumeration, and line credit initialization. During serial preload, the SG2010 does not respond to any transactions on the PCI bus. Additionally, all incoming frames from the link are discarded. If a response frame is required, the Lockout Failure Type is indicated.

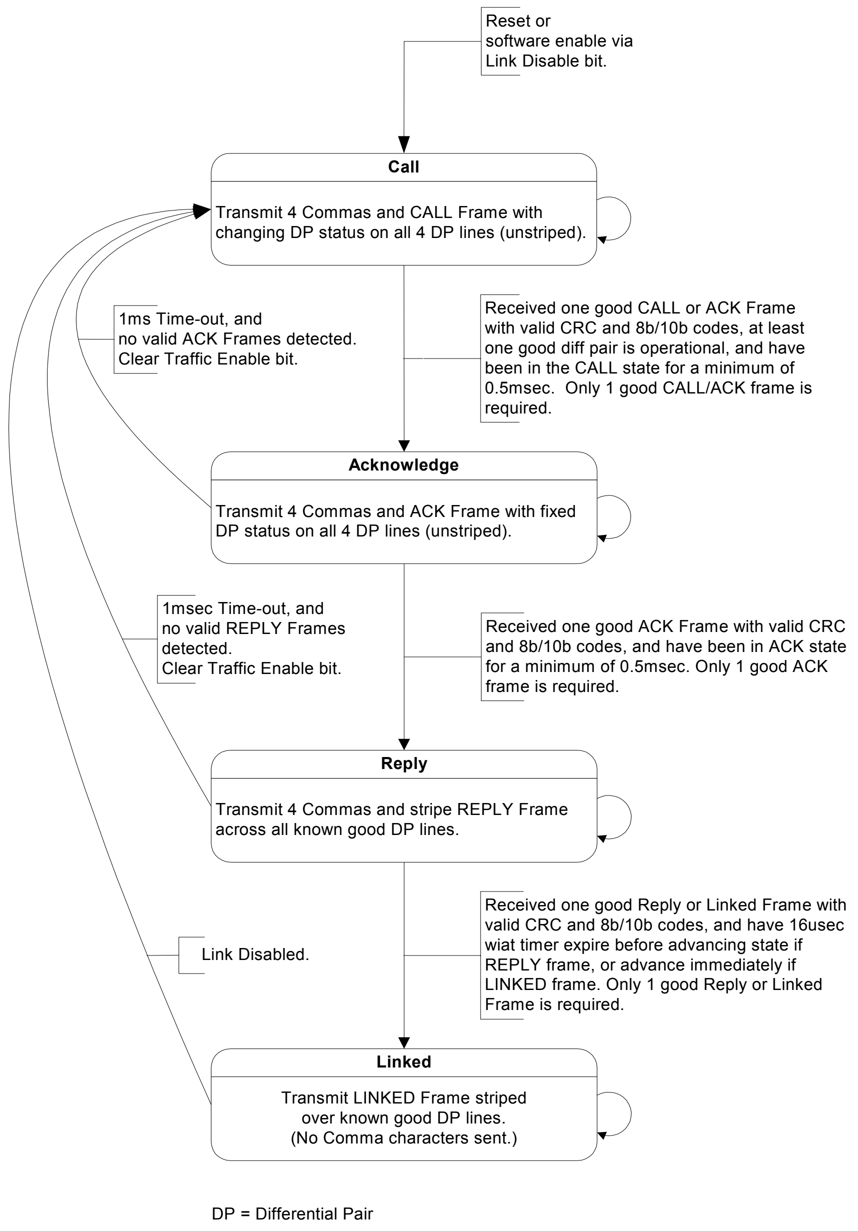
3.8.3 Link Synchronization

After the SG2010 exits reset, it attempts to synchronize its two links with their respective link partners. Each link has its own synchronization state machine. The receiver controls the state machine, which is used by the transmitter for returning status frames.

The four link synchronization states are Call, Acknowledge, Reply, and Linked. When in one of these states, a special frame is transmitted with the proper Behavior field header bits set to indicate the synchronization state. These special frames do not require sequence numbers.

Figure 3–13 shows the link synchronization flow, which is described in the following sections.

Figure 3–13 Link Synchronization State Flow


Reset or
software enable via
Link Disable bit.
Call
Transmit 4 Commas and CALL Frame with changing DP status on all 4 DP lines (unstriped).
1ms Time-out, and no valid ACK Frames detected. Clear Traffic Enable bit.
Received one good CALL or ACK Frame with valid CRC and 8b/10b codes, at least one good diff pair is operational, and have been in the CALL state for a minimum of 0.5msec. Only 1 good CALL/ACK frame is required.
Acknowledge
Transmit 4 Commas and ACK Frame with fixed DP status on all 4 DP lines (unstriped).
1msec Time-out, and no valid REPLY Frames detected. Clear Traffic Enable bit.
Received one good ACK Frame with valid CRC and 8b/10b codes, and have been in ACK state for a minimum of 0.5msec. Only 1 good ACK frame is required.
Reply
Transmit 4 Commas and stripe REPLY Frame across all known good DP lines.
Link Disabled.
Received one good Reply or Linked Frame with valid CRC and 8b/10b codes, and have 16usec wiat timer expire before advancing state if REPLY frame, or advance immediately if LINKED frame. Only 1 good Reply or Linked Frame is required.
Linked
Transmit LINKED Frame striped over known good DP lines. (No Comma characters sent.)
DP = Differential Pair

3.8.3.1 Call State

The Call state is used to achieve byte alignment on each of the four 622Mbs differential pairs in a link. The Call state is entered after the SG2010 exits reset, or after the transmitter is turned on. In the Call state, each 622Mbs differential pair comprising the link continuously sends a special frame with a Behavior field of Call State, with each frame separated by four K28.5 symbols. The SG2010 continues this transmission for a minimum of 31250 62.208MHz clock cycles. If the receiver detects either Call or Acknowledge status with good CRC on any of the four 622Mbs differential pairs in the time period between 31250 and 62500 cycles, then the link state machine assumes byte alignment and transitions to the Acknowledge state. If none of the pairs are receiving Call or Acknowledge status with good CRC in this time period, then the transmitter stops transmitting.

3.8.3.2 Acknowledge State

The Acknowledge state is used to verify reception of good CRC before the transmitter begins to stripe the data over all of the operational differential channels, and to detect which of the differential pairs are working.

Data is still transmitted independently on each differential pair. During the Acknowledge state, the transmitter sends four bytes of K28.5 symbol between Special frames. The Behavior field in the special frame is set to Acknowledge. The special frame also contains information on which differential pairs successfully transmitting good CRC.

The minimum transmission time is 31250 62.208Mhz clock cycles and the maximum time of reception of the acknowledge state from its link partner is 62500 cycles. If a special frame with a Behavior of Acknowledge and good CRC is not received from its link partner during this time, then the transmitter stops transmitting.

If the Acknowledge special frame with good CRC is detected on the link, then the link state machine transitions to the Reply state. If the acknowledge is received earlier than 31250 62.208MHz cycles, then the transition to the Reply state occurs when the 31250 cycle point is reached, but not before, regardless of what is received at that timer expiration point.

3.8.3.3 Reply State

In the Reply state, the transmitter stripes data across 1, 2, or 4 working differential pairs. The transmitter sends four K28.5 symbols between special frames with the Reply state indicated in the Behavior field. The link waits for 62500 62.208MHz cycles to receive the Reply special frame from its link partner with good CRC. If this timer expires before a Reply special frame is received, then the link stops transmitting. If a Reply is successfully received, then the transmitter starts a 1000 62.208MHz cycle timer and moves to the Linked state when this second timer expires. This delay allows sufficient time for its link partner to achieve alignment across its striped channels before the K28.5 symbols are removed. If a Linked frame is received while in the Reply state, the link state machine transitions directly to the Linked state without any time delay.

3.8.3.4 Linked State

In the Linked state, the transmitter sends special frames with the Behavior field set to Linked. This special frame is also known as the Empty frame. This is the state of normal operation for the link, which may transmit data at any time.

3.8.3.5 Disabling a Link

Software can enable or turn off a link through the Link Disable bit in the Link Control and Status register in CSR space (Section 4.6.1.1). The transmitter is always disabled when software brings a link down. It will not transmit even if the receiver detects an incoming link synchronization frame.

However, when the Link Disable bit is 0, the transmitter may be stopped or started depending on the results of link synchronization. If the link fails to synchronize, the transmitter stops, but periodically turns on to attempt to synchronize. The transmitter attempts to synchronize for 1ms every 500ms. In addition, the transmitter turns on and attempts to synchronize if a good Call frame is received from the link partner. If the subsequent synchronization is successful, the link remains enabled. If the synchronization is not successful, it turns off again. After reset the link is automatically enabled and begins the link synchronization sequence.

3.8.4 Fabric Enumeration

Fabric enumeration assigns the port-to-link mapping, Fabric ID, and root port assignment. Fabric enumeration may happen during or after serial preload – there is no timing relationship between them.

Fabric enumeration involves the transmission and reception of special frames. These special frames are:

- I Am frames for identification of the attached component types, bundled links, and ports that are a part of the tree hierarchy
- You Are frames for assignment of the Fabric ID
- Set Credit frames initialize the credit counters of a node’s link partners.

These special frames do not require sequence numbers, nor are credits used to send them. The SG2010 uses transmission timers in order to ensure the transmission of these special frames – one per link for I Am frames, one per link for Set Credit frames, and one for all links for You Are frames. Whenever one of these special frames is sent out any link, the corresponding transmission timer resets and then starts to count. If a transmission error occurs before the timer expires, then all the special frames of that type are re-sent.

To track which type of frame is to be sent out which link, a per-link bitmap is used, one bit for each type of special frame. When any of these bits are set, the appropriate frame is sent out the corresponding link. If multiple bits are set, the following fixed priority is used when sending these frames out a link:

1. Set Credit frame
2. I Am frame
3. You Are frame

There is no ordering relationship between special frames sent out different links.

The following sections describe the various components of fabric enumeration in more detail.

3.8.4.1 Component Identification

Component identification indicates the component type of a node's link partners, as well as the current Fabric ID (FID) of the device. Component identification primarily involves the sending of an I Am special frame. The SG2010 sends an I Am frame out each link when any of the following conditions occur:

- After the link synchronize after a reset deassertion
- When the product of the Traffic Enable bit and the Link Partner Traffic Enable bit transitions from a 0 to a 1, or from a 1 to a 0.
- If the link state transitions to the Linked state after it was in the Call state, and the product of the Traffic Enable bit and the Link Partner Traffic Enable bit is 0 (link up)
- After receiving a You Are frame on any link

When the SG2010 sends an I Am frame, it includes its component type (Edge Node = Yes, Switch = No), the current FID, and the state of its Traffic Enable bit for the link (inverted, in the TDIS field). The Address-Routing Support bit in the I Am frame is set to 0 if the Bridge function is enabled, and to 1 if the Bridge function is disabled. For more information about the FID, see Section 3.8.4.2. Figure 3–14 shows the component identification flow.

3.8.4.1.1 Receiving I Am frames

The SG2010 performs several updates when an I Am special frame is received on a link. The attached component type value is copied into the Link State Table Control and Status register. The Fabric ID and inverted Traffic Disable[6] values are copied into the Link Partner FID register in the corresponding Link State Table entry.

If the FID received in the frame matches an FID received on another link, then the SG2010 bundles the two links, and updates the Port State and Link State tables to reflect the new link to port mapping.

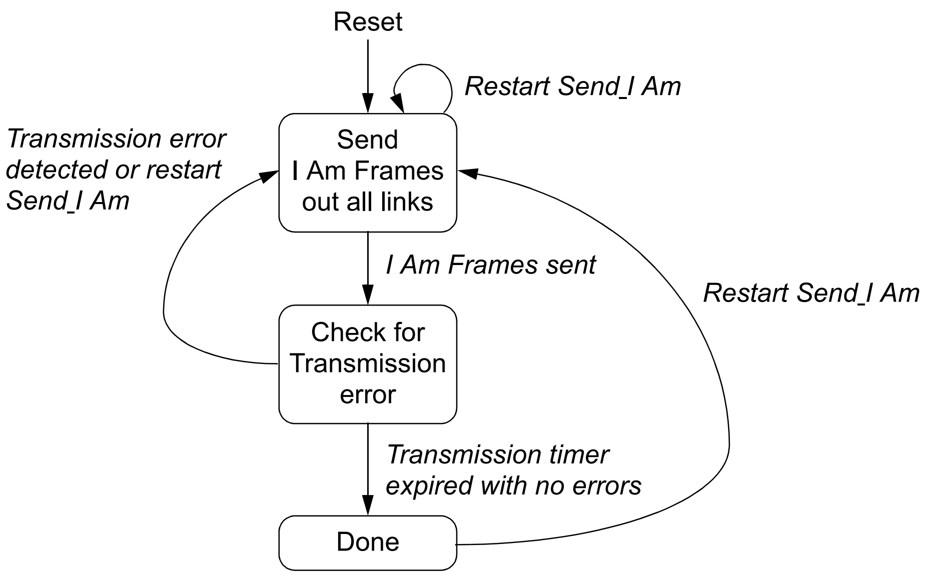
The SG2010 performs a dynamic bundled link check to make sure that the link mapping is not changed after the fabric has been enumerated. If the two links were not previously bundled, the Link Partner FIDs are not 7/7/7777777, and an I Am frame is received on one link that has an FID matching the other link's Link Partner FID, then a bundling error has occurred (formerly unbundled links were bundled). A bundling error code of 001b is indicated in the Link Partner FID register. Similarly, if the two links were bundled, the Link Partner FIDs are not 7/7/7777777, and an I Am frame is received on one link that has an FID that does not match the other link's Link Partner FID, then a bundling error has occurred (formerly bundled links were unbundled). A

6. This inverted value is the Link Partner Traffic Enable bit.

bundling error code of 010b is indicated in the Link Partner FID register. If a bundling error occurs, the Link Partner FID and the attached component type fields are not updated, but the Traffic Disable value is updated.

If the SG2010 is a root, and the FID in the I Am frame matches an FID assigned by the SG2010 in a previously sent You Are frame, then that link, and any bundled links, are considered to be a child of the SG2010 in the fabric tree hierarchy, and the Port Map Table Enable bit for that port is set if the Bridge function is enabled. The Port Map Table Enable bit is changed upon reception of an I Am frame only when the Traffic Enable bit is set for the link where the I Am frame was received.

Figure 3–14 Send I Am Component Identification


Reset
Restart Send_I Am
Transmission error detected or restart Send_I Am
Send I Am Frames out all links
I Am Frames sent
Restart Send_I Am
Check for Transmission error
Transmission timer expired with no errors
Done

3.8.4.2 Fabric ID Assignment

This part of fabric enumeration assigns an FID to every node and also constructs a tree hierarchy within the fabric to be used for address-routed traffic. Fabric enumeration is initiated by the root or through a software generated frame to a link partner and can happen at any time after link synchronization. The root determines whether enumeration is performed, and if so, the timing of fabric enumeration. When the SG2010 is the root, it starts fabric enumeration after performing component identification.

When the SG2010, as a root, sends the first You Are frame to a link partner, it starts a fabric enumeration timer. Through serial preload, this timer can be programmed to expire after 2.5ms, 5ms, 10ms, or 20ms. The SG2010 does not allow frames to be sent into the fabric until after the enumeration timer expires – these transactions result in a master abort on the PCI bus. Access is allowed to all of SG2010’s registers (as long as serial preload is not ongoing; in that case they would terminate in master abort).

The FID has the following components:

- Parallel fabric number (PFN) – used when two or more nodes are connected to the root, essentially forming parallel tree hierarchies. Bit 0 of the PFN used by the SG2010 (as a root) is controlled by a strapping pin.
- Turn count – incremented at each switch similar to the turn count of a path. All devices with the same turn count will also have the same PCI bus number (although the turn count is not necessarily equal to the bus number). The turn count indicates the number of valid turns in the FID.
- Turns – uniquely identifies the downstream link partners with the same Turn Count.

These components of the FID are identified using the nomenclature:

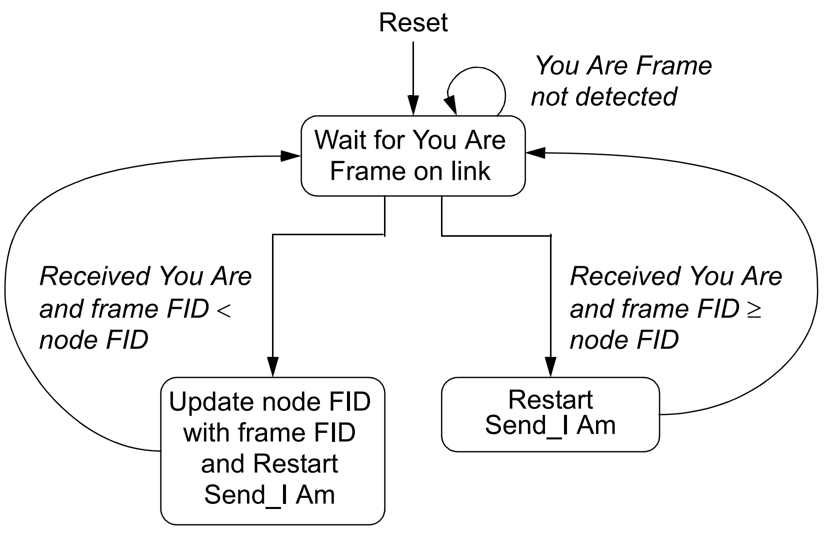
PFN/turn count/turns

During reset, if the SG2010 is a:

- Leaf, the FID is initialized to 7/7/7777777
- Root, and PFN[0] is strapped to:
  - 0, the FID is initialized to 0/0/7777777
  - 1, the FID is initialized to 1/0/7777777

Figure 3–15 shows the FID assignment flow for a leaf.

Figure 3–15 Leaf Fabric ID Assignment Flow


Reset
You Are Frame not detected
Wait for You Are Frame on link
Received You Are and frame FID < node FID
Received You Are and frame FID ≥ node FID
Update node FID with frame FID and Restart Send_I Am
Restart Send_I Am

When the SG2010 is a leaf, the FID assignment starts when a You Are special frame is received on one of SG2010's links. The SG2010 compares the FID received in the frame to its current FID. If the received FID is less than the current FID, then the SG2010 updates its FID with the new one. The SG2010 performs the comparison and conditional FID update in the following order:

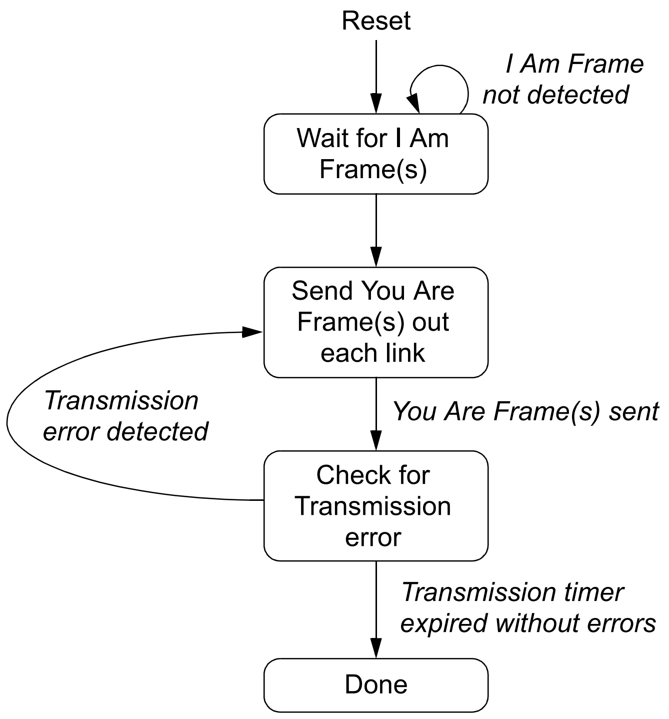
- Address-routing bit comparison
  - The SG2010 updates the FID based on the lowest address-routing bit received in a You Are frame

  - The SG2010 prefers a parent that supports address routing
- Parallel Fabric Number (PFN) comparison
  - If new PFN < current PFN, the FID is updated and no further comparisons are necessary
  - If new PFN > current PFN, the FID is not updated and no further comparisons are necessary
  - If new PFN = current PFN, then the turn count is compared
- Turn Count comparison
  - If new turn count < current turn count, the FID is updated and no further comparisons are necessary
  - If new turn count > current turn count, the FID is not updated and no further comparisons are necessary
  - If new turn count = current turn count, then the turn value is compared
- Turn comparison (that is, turn *x* in the FID where *x* = (turn count – 1))
  - If new turn < current turn, the FID is updated
  - If new turn ≥ current turn, the FID is not updated

If a leaf, when the SG2010 updates its FID based on the FID received in a You Are frame, it sets the receiving port to be the root port. Regardless of whether the FID is updated, when a You Are frame is received, the SG2010 restarts the Component Identification state machine to inform all of its link partners with its current identity (FID).

Figure 3–16 shows the FID assignment flow for a root.

Figure 3–16 Root Fabric ID Assignment


Reset
I Am Frame not detected
Wait for I Am Frame(s)
Send You Are Frame(s) out each link
Transmission error detected
You Are Frame(s) sent
Check for Transmission error
Transmission timer expired without errors
Done

A root SG2010 assigns FIDs to its link partners by sending h a special frame with the You Are command. The product of the Traffic Enable (TEN) and the Link Partner Traffic Enable (LP_TEN) bits must be a 1 in order to send a You Are frame out a link. The SG2010 sets the address-routing bit for the You Are frame to 0 if the Bridge function is enabled; 1 if disabled. The PFN for each link is chosen based on the attached component type and the link number. Bit [0] of the PFN is selected based on the state of the strapping pin at reset (high = 1, low = 0). Bit [1] of the PFN corresponds to the link number (0 or 1). Bit [2] of the PFN is 0 if the link partner is a switch, or 1 if the link partner is an edge node. Table 3–34 lists possible PFN combinations.

Table 3–34 PFN Assignment During Fabric Enumeration

Link	Link Partner Component Type	PFN	
–	–	PFN[0]=0	PFN[0]=1
Link 0 Link 1	Switch Switch	0 2	1 3
Link 0 Link 1	Switch Edge Node	0 6	1 7
Link 0 Link 1	Edge Node Switch	4 2	5 3
Link 0 Link 1	Edge Node Edge Node	4 6	5 7
Link 0 Link 1	Switch Not synchronized	0 N/A	1 N/A

Table 3–34 PFN Assignment During Fabric Enumeration

Link 0 Link 1	Not synchronized Switch	N/A 2	N/A 3
Link 0 Link 1	Edge Node Not synchronized	4 N/A	5 N/A
Link 0 Link 1	Not synchronized Edge Node	N/A 6	N/A 7

When the SG2010 is a root, there is no root port because the PCI bus acts as the root port. Additionally, a different value cannot be assigned to SG2010's FID when the SG2010 is the root.

Note that this document covers SG2010's implementation of the fabric enumeration process. For a complete description of the fabric enumeration process, see *The StarFabric Architecture Specification.*

3.8.4.3 Using the Fabric ID as a Path to the Root

When the SG2010 is a leaf, its FID can be manipulated to obtain a path to the root using the turn count and seven turn values of the FID. Each of these components is a three-bit octal number. To derive the path back to the root, the active turns are reversed and inverted (path transform). The turn count indicates the number of active turns, where an FID turn count of 4 indicates four active turns. The output port is not reflected in the path transform. The Root bit in the Port State Table determines the output port towards the root.

Figure 3–17 is an example of how a path to the root is created from the FID. In this example, there are four active turns. A path transform is performed on the first four turns (FID turns 7 0 2 4 are transformed into 3 5 7 0). The remaining inactive turns remain as 0.

Figure 3–17 Root Path Example

Fabric ID Format	FID Turn 6	FID Turn 5	FID Turn 4	FID Turn 3	FID Turn 2	FID Turn 1	FID Turn 0	Turn Count	PFN	RES
Fabric ID Example	0	0	0	7	0	2	4	4	0	RES
Path to Root	0	0	0	3	5	7	0	0		

3.8.4.4 Line Credit Initialization

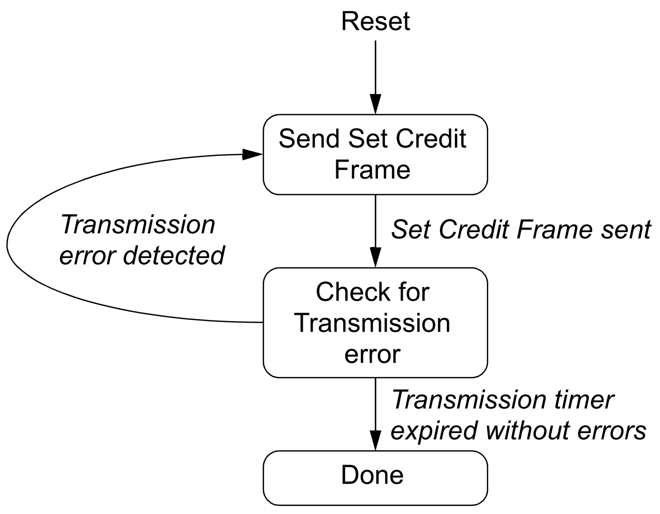
After reset and link synchronization, the SG2010 initializes the line credit counters of its link partners. SG2010 link partners use these credits when sending credited frames to the SG2010.

The SG2010 sends a CoS Set Credit special frame containing class-of-service line credit information out each newly synchronized link. The SG2010 does not sent a Turn Set Credit special frame; in order to enable turn credits for SG2010 buffers, credit reallocation must be performed by software.

The SG2010 also sends a CoS Set Credit frame when the product of its Traffic Enable and Link Partner Traffic Enable bits for that link transitions from a 0 to 1.

Figure 3–18 shows the set credit flow.

Figure 3–18 Figure 17 Set Credit Flow


Reset
Send Set Credit Frame
Transmission error detected
Set Credit Frame sent
Check for Transmission error
Transmission timer expired without errors
Done

The link partner loads its CoS credit counters with the values contained in the Set Credit frame. Table 3–35 lists the number of credits assigned per link at reset.

Table 3–35 Initial Line Credits for SG2010

Credit Type	Write/Response Credits	Request Credits
Asynchronous	0	0
Isochronous	46	12
HP-Asynchronous	0	0
Multicast	46	–
Address-routed	68	12
HP-Isochronous	0	0
Provisioning	56	12

The SG1010 also receives Set Credit special frames from its link partner(s) on each synchronizing link. The link partner must send a Cos Set Credit frame, and may optionally send a Turn Set Credit frame. The SG1010 initializes its line credit counters for each link with this information.To support CoS line credit aliasing, when the SG2010 receives the Set Credit frame, it adds together the credits from the following CoS pairs and loads only the counter of the first CoS:

- Address-routed and asynchronous

  - Result loaded into address-routed credit counters
- Isochronous and HP-isochronous
  - Result loaded into isochronous credit counters
- Provisioning and HP-asynchronous
  - Result loaded into provisioning credit counters

3.8.5 PCI Configuration

After serial preload completes and the Fabric Enumeration Timer expires, the SG2010 as the root can accept PCI configuration transactions from the host. A host performs standard PCI configuration of the fabric as it would any PCI hierarchy. Configuration transactions are propagated through the fabric as address-routed read request and write frames. A leaf SG2010 can accept configuration read and write frames as soon as serial preload is complete and the links have initialized.

3.8.5.1 Port Map Table Initialization

The SG2010 Port Map Table contains PCI address-routing decode ranges for its link partners. The SG2010 requires this information to determine which port to use when forwarding a PCI address-routed frame through the PCI hierarchy. To transparently forward PCI frames through the PCI hierarchy, nodes must automatically store or distribute this information.

When the SG2010 is a root, it snoops configuration writes to configuration registers when one of its downstream port partners is the target. This occurs when the SG2010 sends a Type0 configuration write frame out the port. The SG2010 captures the write data and copies it into the corresponding Port Map Table register.

When the SG2010 is a leaf, it does not have any downstream fabric devices so does not perform snooping. However, in the three-bridge configuration, the SG2010 must have the Port Map Table information for the peer downstream bridge. When the SG2010 is a leaf, it exchanges Port Map Table update information with its link partners using a provisioning Channel 255 write frame specifying Port Map Table update (offset[35]=1). The lower three offset bits specify the Port Map Register offset within the Port Map Table. When the SG2010 receives a Port Map Table update frame, it updates the Port Map Table register corresponding to the given offset and input port with the write data.

The SG2010 always sends a Port Map update write frame when one of the configuration registers specified above is written with a configuration write operation. If the SG2010 is configured to have two ports, then two frames are sent. If a link partner is relying on snooping to update its Port Map registers, it may simply discard the frame.

If the above configuration registers are updated through another means, such as Channel 255 write, or PCI memory or I/O write, snooping is not performed, and the Port Map update write frames are not sent. In this case, software must insure that the Port Map Tables are consistent with the link partner's configuration registers.

3.8.5.2 Lockout

To prevent address-routed fabric access to SG2010 registers and the secondary PCI bus until local PCI initialization is performed, the SG2010 implements a lockout mechanism that can be used when it is a leaf. The SG2010 implements a Lockout bit in the Chip Control configuration register in the Gateway function. This bit is initialized through a strapping pin, and may also be written through a serial ROM preload. Serial preload will overwrite the value set by the strapping pin. As long as this bit is set to 1, the SG2010 responds to address-routed frames with a Read Completion or Write Acknowledge frame containing Lockout status in the Failure Type. Frames not requiring a write acknowledge or read completion are discarded.

If the Lockout bit is 0, the SG2010 responds normally. When local initialization is complete, software or serial preload should clear the Lockout bit to 0. Software can clear, but not set the Lockout bit.

3.9 Link and Port Operation

3.9.1 Link and Port Conditions

Port and link state is reflected in the Port State Table and Link State Table registers, respectively. Port state is a logical condition and indicates whether a port can transmit credit-based traffic. Link state is a physical condition and indicates how many of its transmit and receive differential pairs are synchronized.

In order for traffic to be sent out a link, both the Traffic Enable (TEN) and the Link Partner Traffic Enable (LP_TEN) bits for that link must be 1. In other words, the product of TEN and LP_TEN must be 1. The TEN bit is located in the Link State Table Control and Status register. The LP_TEN bit is located in the Link Partner Fabric ID register. Only link synchronization frames, I Am frames, and Set State frames may be sent when the product of TEN and LP_TEN is 0.

3.9.1.1 Link Up

A link is up when at least one transmit and receive differential pair is synchronized. A link may also be fragile if less than four transmit and receive differential pairs are synchronized.

When a link goes from the Call synchronization state to the Linked synchronization state, and the product of the TEN bit and the LP_TEN bit is 0, the following flow occurs:

- The SG2010 sends an I Am frame to its link partner
- The SG2010 receives an I Am frame from its link partner
  - The Fabric ID in the frame is stored in the Link Partner Fabric ID register
  - The TDIS bit is inverted and stored in the LP_TEN bit in the Link Partner Fabric ID register

  - Reception of the I Am frame does not affect the current values of the Port Map Table Enable in the Port State Table if the TEN bit is clear
- After sending the I Am frame, the SG2010 signals a Link Up event

Software must then intervene to bring the link the rest of the way up by writing the TEN bit in either the SG2010 or the link partner. If software writes the TEN bit in the SG2010, the SG2010 sends a special Set State frame to its link partner with Set_TEN=1. Otherwise, if software writes the TEN bit in the link partner, the link partner sends the special Set State frame with Set_TEN to the SG2010.

If the SG2010 receives a Set State frame with Set_TEN=1, then the SG2010 sets both its TEN bit and the LP_TEN bit for that link. Upon reception of the Set State frame and setting of these bits, the SG2010 returns an I Am frame to the link partner with TDIS=0 (that is, TEN is 1).

If the SG2010 sent the Set State frame, then the link partner sends an I Am frame back to the SG2010. The TDIS bit should be 0, indicating that the link partner's TEN bit is set. The SG2010 then updates the Link Partner Fabric ID register with both the Fabric ID of the frame and the inverted TDIS bit to indicate the state of LP_TEN.

When both the Traffic Enable bit and the LP_TEN bits are 1, the SG2010 sends a Set Credit frame to its link partner with default credit values. The link is now ready to send and receive traffic. Software must ensure that both TEN and LP_TEN bits are 1 before it considers the link to be fully operational.

When links are bundled, and software writes the TEN bit of one link, the TEN bit of the other link is also set. Set State and Set Credit frames are sent out both links in the bundle as described above.

Software may attempt to bring a link up from a down state by writing a 0 to the Link Disable bit, which turns on the link's transmitter. If the link synchronizes successfully, a Link Up event is signaled and the link up process described above must be followed.

3.9.1.2 Link Down

A link down occurs when the product of the Traffic Enable bit (Link State Table Control and Status register) and the Link Partner Traffic Enable bit (Link Partner FID register) transitions from a 1 to a 0. This may happen in the SG2010 for the following reasons:

- The link is attempting to synchronize, and synchronization fails during either the Call, Ack, or Reply state (See Figure 3–13)
- An I Am frame is received where TDIS=1, causing the Link Partner Fabric ID bit to be cleared
- An I Am frame is received with an FID of 7/7/7777777, and the current Link Partner FID register is not 7/7/7777777 (indicates a link partner reset)
- Software clears the Traffic Enable bit

  - When links are bundled, clearing one Traffic Enable bit causes the other link's Traffic Enable bit to also be cleared, causing the entire port to go down
- Software sets the Link Disable bit, turning off the link's transmitter. This causes hardware to clear the Traffic Enable bit due to loss of synchronization
- The Maskable Reset Mode bit is a 1 in the Chip Control register in Gateway Configuration space, and a maskable reset comma is received

When a link down occurs, a Link Down chip event is signaled. If the link down is not due to failed synchronization, the SG2010 forces the link state to transition to the Call state. The link then attempts to resynchronize. If the resynchronization is successful, the SG2010 signals a Link Up event as describe in Section 3.9.1.1. Additionally, SG2010 hardware resets the credit counters to their default values in the corresponding Link State Table. The SG2010 also resets the sequence numbers to 0 for that link. If the links resynchronize, the SG2010 sends an I Am frame to its link partners.

3.9.1.3 Fragile Links

Fragile link state is a physical statement about the number of working differential pairs in a given link. A link is fragile when any one or more of its differential receive or transmit pairs lost or could not achieve synchronization, but at least one differential transmit and receive pair is working. The SG2010 supports configurations of 1, 2, or 4 working transmit and receive pairs; a fragile link has 1 or 2 working differential pairs. The Differential Pair State register in the Link State Table shows which differential transmit and receive differential pairs are synchronized and which are not.

When a link transitions to a fragile state, a CRC or 8B/10B transmission error will be detected and will eventually cause the link to resynchronize. Upon resynchronization, the SG2010 detects that the number of working differential pairs has been reduced and signals a Fragile Link chip event. However, traffic is still transmitted and received across that link using the existing synchronized pairs.

When a link is fragile, the receiver for the non-working differential pairs remains on. The transmitter attempts to synchronize for 1ms every 500ms. If the SG2010 detects that a differential pair has become operational, it automatically forces a resynchronization on the link to include the new differential pair as part of the link.

If the links are bundled and one of the links goes fragile while the other link is operating at full speed, the SG2010 clears the Traffic Enable bit of the fragile link so that it cannot transmit routed frames. If both links go fragile, the SG2010 takes down the most fragile link, or Link 0 if they are at the same level of fragility.

3.9.1.4 Port Up

A port is up when at least one of its links has its corresponding Traffic Enable bit set. A port is not considered to be up until it can transmit credit-based traffic. Since software must intervene to bring a port from a down state to and up state, an event is not signaled when a port goes up. When a port comes up after reset, software intervention is not required and an event is not signaled.

3.9.1.5 Port Down

A port is down when the Traffic Enable bits of all of the links that comprise it are clear. When a port is down, none of its links can transmit traffic. The Port State Table contains a Port Up/Down State bit that reflects the state of the corresponding port. A Port Down event is signaled when it goes from an up state to a down state (the product of TEN and LP_TEN for all links in the port are 0).

Note that if a port is down but at least one of its links is synchronized, it can still transmit and receive I Am, Set State, and link synchronization special frames even if the product of TEN and LP_TEN is 0. However, it cannot transmit any other frames.

3.9.2 Sending Frames

Frames can only be transmitted by a link when it is in the Linked synchronization state. Additionally, the Traffic Enable bit and the LP_TEN (Link Partner Traffic Enable) bits must both be set in order to send frames, with the exception of I Am and Set State frames. These frames must be used to bring a link up to an operational state after the link goes down. When a frame is sent, the sequence number link overhead byte is sent first, followed by the frame itself, the line credit return byte, and two bytes of CRC.

Frame lines are sent in order starting with line 0. Within a line, byte 0 is sent first. Within a byte, the most significant bit of the 8B/10B encoded byte is sent first and the least significant bit is sent last.

3.9.2.1 Empty Frames

Empty frames (Special Link Synchronization frames) are sent when there are no other frames to send. The empty frame contains the synchronization status of the link as well as which of the differential pairs are operational.

3.9.2.2 Sequence Numbers

Sequence numbers are used to order frames that are received on a bundled port. Additionally, they are used to indicate which frames need to be re-sent when an 8B/10B or CRC transmission error occurs.

All credited frames and the special line credit update frames use sequence numbers. Other special frames do not use sequence numbers.

A sequence number is a seven-bit field that is prepended to a frame as a part of its link overhead. Bit [7] contains the Thread ID, bits [6:1] contain the Frame Number, and bit [0] contains the Line Debit Type, the latter indicating whether CoS or turn credits were used to send the frame.

If a port has only one link, the Thread ID is always 0; if it has two links, the Thread ID can be 0 or 1. A Thread ID of 0 indicates that the frame is dependent on the frame previously sent on the same link. When the Thread ID is 0, the Frame Number is the current value of the frame counter for that link (also called the raw frame number). A

Thread ID of 1 indicates that the frame is dependent on a frame sent out the other link in the bundle; in this case, the Frame Number is the raw frame number of the frame it is dependent on.

The SG2010 uses these sequence numbers to order frames received on different links in a bundled port. If the port is not bundled, the frames are always received in order. For more information about sequence numbers, see the *StarFabric Architecture Specification*.

When a link goes down and comes up, the link partners must reset their sequence numbers in order for the sending and receiving of frame traffic to resume. There are two sets of sequence numbers for each link - the transmit sequence numbers which are used in the link overhead of sent frames, and receive sequence numbers, which are the expected sequence numbers of frames that come into the link. Both must be reset. Sequence numbers are reset when both the Traffic Enable bit and the LP_TEN bit are both 1, where one or both were previously 0.

3.9.2.2.1 Using Change Thread Frames When a Link Goes Down

When a bundled link loses synchronization or there is a transmission error, the SG2010 resends all the frames that could have been affected by the error out the other link. Before the SG2010 resends these frames, it first waits until the frames transmitted on the other link have been received without error, and then sends a Special Change Thread frame out the working link to tell the receiver that the thread has been temporarily changed to correspond to the link that had the error, and sends the same Frame Number as the first frame to be re-sent. The SG2010 uses a timer to determine that the Special Change Thread frame is successfully sent. After this timer expires and no transmission errors have been detected, the SG2010 resends the frames. After the frames have been successfully re-sent, the SG2010 sends another Special Change Thread frame to change the thread back to its original state.

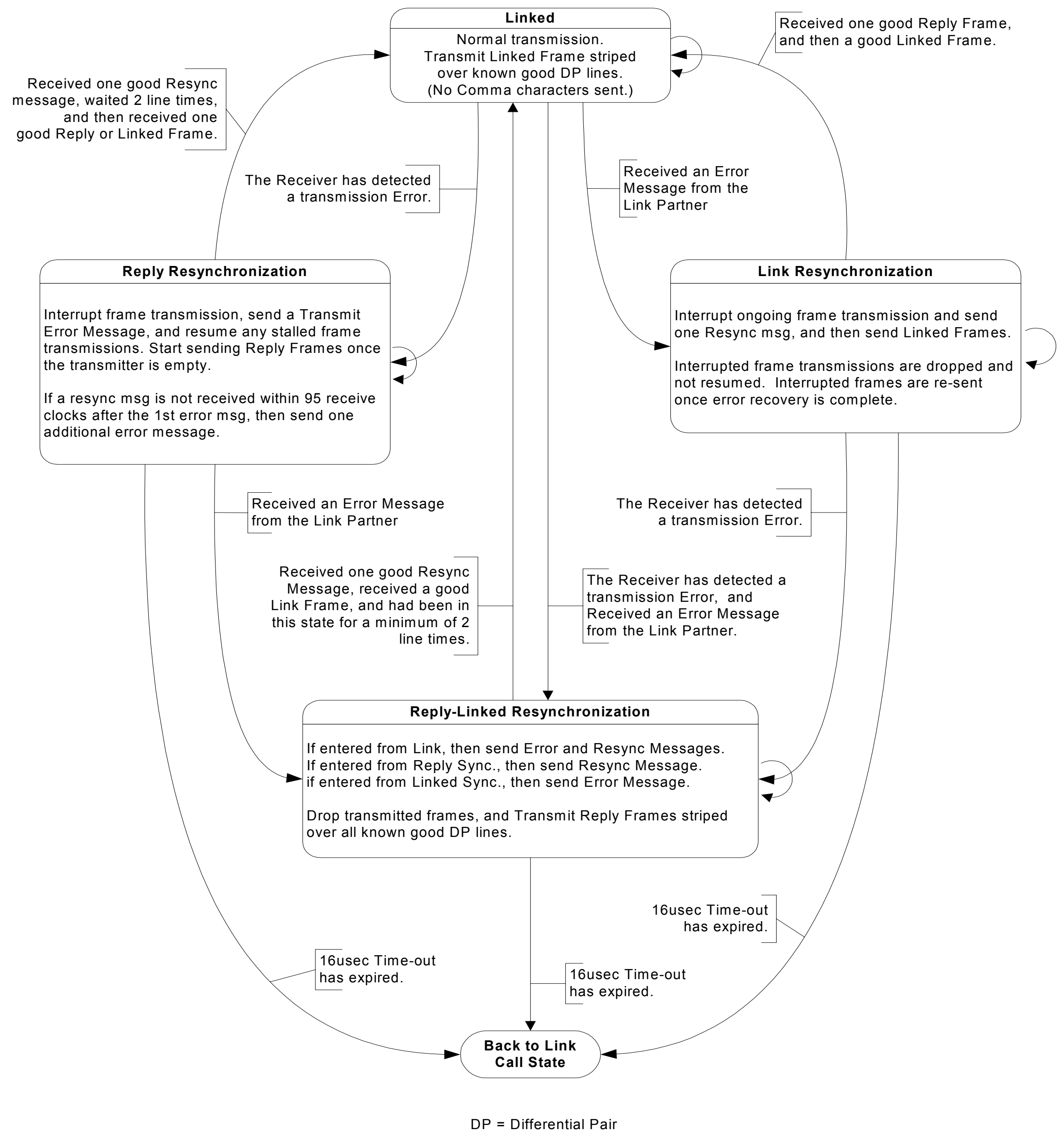
3.9.2.3 Transmission Errors

Data integrity is tracked using two mechanisms – 8B/10B encoding and CRC error checking. 8B/10B encoding covers each byte of data sent. CRC checking covers the entire frame, including link overhead (sequence number and line credit byte). StarFabric protocol uses a 16-bit CRC polynomial. 8B/10B and CRC transmission error recovery are treated in the same way. The transmission error recovery flow is shown in Figure 3–19 and is described in the following sections.

An *error message* is 4 bytes of data sent by the node detecting the transmission error to the node that sent the erroneous frame. The error message contains a K28.5 comma character, followed by a byte containing the sequence number of the frame in error and the error type indication (8B/10B or CRC), followed by a repeat of that byte, followed by another K28.5 comma character. The error message is sent on all working differential pairs; it is not striped. An error message is successfully received when a node detects the error message on at least one working pair. The error message is not covered by CRC, however the receiver of the error message must determine that the two data bytes in the message are identical.

A *resync message* consists of four K28.5 comma characters in a row transmitted across each differential pair. A resync message is sent by a node that detects an error message, and is used to acknowledge the reception of the error message.

Figure 3–19 Transmission Error Recovery State Flow


Linked
Normal transmission. Transmit Linked Frame striped over known good DP lines. (No Comma characters sent.)
Received one good Reply Frame, and then a good Linked Frame.
Received one good Resync message, waited 2 line times, and then received one good Reply or Linked Frame.
The Receiver has detected a transmission Error.
Received an Error Message from the Link Partner
Reply Resynchronization
Interrupt frame transmission, send a Transmit Error Message, and resume any stalled frame transmissions. Start sending Reply Frames once the transmitter is empty.
If a resync msg is not received within 95 receive clocks after the 1st error msg, then send one additional error message.
Link Resynchronization
Interrupt ongoing frame transmission and send one Resync msg, and then send Linked Frames.
Interrupted frame transmissions are dropped and not resumed. Interrupted frames are re-sent once error recovery is complete.
Received an Error Message from the Link Partner
The Receiver has detected a transmission Error.
Received one good Resync Message, received a good Link Frame, and had been in this state for a minimum of 2 line times.
The Receiver has detected a transmission Error, and Received an Error Message from the Link Partner.
Reply-Linked Resynchronization
If entered from Link, then send Error and Resync Messages.
If entered from Reply Sync., then send Resync Message.
if entered from Linked Sync., then send Error Message.
Drop transmitted frames, and Transmit Reply Frames striped over all known good DP lines.
16usec Time-out has expired.
16usec Time-out has expired.
16usec Time-out has expired.
Back to Link Call State
DP = Differential Pair

3.9.2.3.1 Detecting a Transmission Error

When the SG2010 detects an 8B/10B or CRC transmission error on received data, it marks the current line as end of frame. Data received on that link after the error is detected is dropped. The SG2010 sends an error message to its link partner indicating the type of error that was detected (8B/10B or CRC), whether the error occurred on that link or on a bundled link, and the current received frame count (this count is initialized by the link partner when an I Am frame is received on the link and incremented every time a frame is received). If the port is bundled, the error message is sent out the other link in the port. Otherwise, it is sent out the same link that received the error. The SG2010 may interrupt frame transmission to send the error message, and then resume that frame transmission after the error message is successfully sent.

The SG2010 transitions the link that received the frame with error to the Reply state. If a resync message is not received within 95 62.208MHz clock cycles, then the SG2010 sends another copy of the error message and starts a 16µsec timer. While the SG2010 is waiting for a resync message it ignores any other data received on that link. When the resync message is received, the transmitter transitions the link back into the Linked state and resumes normal operation. If the 16µsec timer expires and the resync message was not detected, then the SG2010 transitions that link into the Call state for a full resynchronization as described in Section 3.8.3.

3.9.2.3.2 Receiving a Transmission Error Message

The SG2010 receives a transmission error message from a link partner when one of the frames it send to the link partner contained an 8B/10B or CRC error when it was received.

Because error messages are not striped across differential pairs, identical error messages are sent on each working differential pair. The SG2010 must detect a valid error message on one or more differential pairs to start error recovery. A valid error message has two identical copies of the error byte and no 8b/10b errors in any of the four bytes. The receiver drops to the Call state only if it receives invalid error messages on all operational differential pairs.

The error message interrupts the regular stream of frames. The SG2010 does not drop any frames received during or after the reception of an error message.

When the SG2010 receives an error message from its link partner indicating that an 8B/10B or CRC error was detected, it sends a resync message out the link where the error message was received to acknowledge reception of the error message and to transition the link partner into the Linked state. Once the SG2010 received Linked state special frames from its link partner, it may re-send the frames.

The SG2010 determines on which link the error occurred from the error message. It detects the first frame that must be re-sent from the sequence number received in the error message. If the port is bundled, the SG2010 re-sends the frames out the other link as described in Section 3.9.2.2.1. Otherwise, if there is only one link in the port, the frames must be re-sent out the same link. All frames from the frame in error to the last frame sent are re-transmitted.

3.9.2.3.3 Kill Frame Credit Byte Encoding

The buffer credit byte appended to a transmitted frame as part of the link overhead has a “Kill Frame” encoding as one of its possible meanings. When this encoding is received, the frame that is attached to that buffer credit byte is discarded, much as if a transmission error was detected on the frame. When the SG2010 discards a frame due to a Kill Frame encoding, the SG2010 expects the next frame that is transmitted to have the same sequence number as the dropped frame.

No error message is sent to the node that transmitted the frame in this case, and no event is signaled.

3.10 CompactPCI Hot Swap

The SG2010 implements CompactPCI Hot Swap functionality that allows a Compact-PCI card with an SG2010 on it to be inserted into and removed from an operating PCI bus backplane. The support includes a hot swap pin interface, a configuration register software interface, a hot swap state machine, and electrical compatibility. All hot-swap-related state, including architected and device-specific register bits, and the hot-swap controller state machine are reset only on the assertion of LRST_L or RST_L. Resets initiated by software through setting a bit and propagated resets received on the links do not reset hot swap functionality.

The SG2010 meets all requirements to be hot-swap capable and hot-swap friendly. Additionally, it provides the hot swap feature of local 64-bit initialization support.

3.10.1 Hot Swap Pin Interface

Table 3–36 lists the hot swap interface pins implemented by the SG2010.

Table 3–36 CompactPCI Hot Swap Signal Pins

Name	Type	Description
LSTAT	I	Hot swap local status. This input indicates the status of the Compact PCI card ejector handle (switch). When 1, the switch is open or unlatched. When 0, the switch is closed or latched. The SG2010 debounces this signal to ensure that a valid transition is detected. The debounced value of LSTAT is reflected in the Chip Status register in Gateway configuration space.
HS_LED	O	LED Control. This output controls the LED associated with the hot swap function.
LRST_L	I	Hot swap local reset. This input is asserted low by external logic when the CompactPCI card is inserted. Causes a full reset as if the PCI global reset signal was asserted.
ENUM_L	OD	Hot swap interrupt signal. Asserted low to the processor to indicate card insertion or impending removal, so that the processor can initialize or quiesce software, respectively.
L64EN_L	I	Hot swap local 64-bit enable. This input is sampled when LRST_L is asserted to determine whether the PCI interface is connected to a 64-bit bus or 32-bit bus.
BDSEL_L	I	Board seated signal. When high, the SG2010 will not respond to nor initiate any PCI transactions. When low, the SG2010 responds to and initiates PCI transactions normally. The value of BDSEL_L is reflected in the Chip Status register in Gateway configuration space.

3.10.2 Hot Swap Register Interface

The SG2010 implements the Hot Swap Control register in configuration space. When the Bridge function is enabled, the register appears in Bridge configuration space; when the Bridge function is disabled, the register appears in Gateway configuration space. Table 3–37 describes the control and status bits defined in the Hot Swap Control register.

Table 3–37 Hot Swap Control Register Control and Status Bits

Bit Name	Description
DHE	Device Hiding Enable. When 1, device hiding, as controlled by the LOO bit, is enabled. When 0, device hiding is disabled. When the PI bit is 0, this bit is read-only as 0.
EIM	ENUM interrupt mask. When 0, ENUM_L is asserted low whenever INS or EXT are set. When 1, ENUM_L is always tristated. Reset: 0.
PIE	Pending Insertion/Extraction. When 1, an insertion or removal is in progress. When 0, the board is installed in view of hardware and software. When the PI bit is 0, this bit is read-only as 0.
LOO	LED on/off. When the LED is under software control and LOO is 1, the HS_LED signal is asserted and the LED is illuminated. When the LED is under software control and LOO is 0, the HS_LED signal is deasserted and the LED is off.
PI	Specifies the hot swap programming interface. The default value is 1 (DHE and PIE implemented), but this bit can be loaded by SROM preload to revert to version 0.
EXT	Pending extraction status bit. Set by the SG2010 when the switch is opened (LSTAT = 1) and the board is in the Installed state. EXT is set by an edge detection on LSTAT to prevent accidental setting of this bit. Cleared when software writes 1 to this location. Software cannot write this bit to 1.
INS	Board inserted status bit. Set by the SG2010 when the device is ready for configuration, after switch has been closed (LSTAT = 0), reset has been deasserted, and any local initialization has been completed. Software clears this bit by writing 1 to this location. Software cannot write this bit to 1.

Additionally, in the Chip Control register in Gateway Configuration space, the SG2010 implements two device-specific bits that affect the hot swap controller:

Bit	Description
Skip INS Enable	When 1, enables the hot swap controller to skip insertion state after power up. It does not effect re-insertion after removal. When 0, the hot swap controller goes through the insertion state as specified in the CompactPCI Hot Swap specification. The reset value of this bit is determined by a strapping pin.
Remote INS Ready	Only meaningful in the H/W Disconnected state when the SG2010 is configured to assert ENUM_L remotely. Remote ENUM_L is enabled when the ENUM_L Input Signal Mask is disabled (0) in the Event Dispatch Control register. When Remote INS Ready is a 1, is indicates that the SG2010 is ready to forward signal event frames and the hot swap controller can leave the H/W Disconnected state. When 0, and a remote ENUM_L is indicated and insertion is not skipped, the SG2010 remains in this state until one of these conditions changes. Reset to 0.

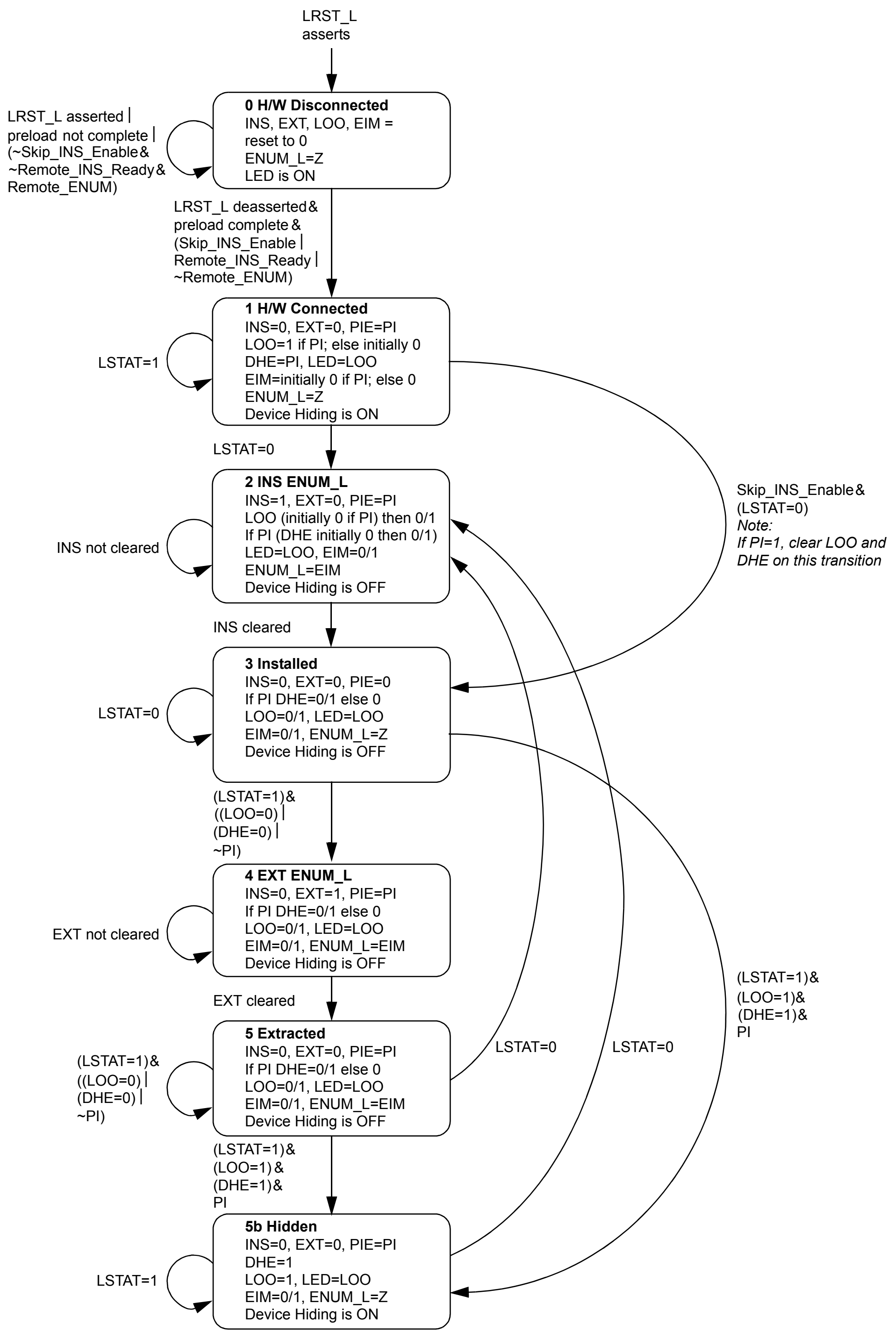
The values of both of these bits can be programmed through serial preload. If the SG2010 is in device hiding mode, the bits can still be accessed from the StarFabric interface.[7]

3.10.3 Hot Swap Controller Implementation

Figure 3–20 is the hot swap controller state diagram. Sections 3.10.3.1 through 3.10.3.7 describe each state and its transitions.

7. When ENUM_L is remote (signal event frame generated), at least one of these two bits must be set to 1 either through serial preload or fabric maintenance software. Otherwise the hot swap state machine remains in State 0, where the SG2010 is inaccessible from the PCI interface. Remote ENUM_L is specified by default when the SG2010 is a leaf and the Bridge function is enabled.

Figure 3–20 Hot Swap Controller Diagram



PI=preloadable; default=1
Handle Open=(LSTAT=1)
Handle Closed=(LSTAT=0)
LED On=(LED=1)
LED Off=(LED=0)
Device Hiding=(State 0 | State 1 | State 5b | (BDSEL_L=1)) & (PI=1)

3.10.3.1 State 0 H/W Disconnected

The hot swap controller enters state 0 when LRST_L is asserted, regardless of the previous controller state. If the board has just been inserted, only early power may be applied. LRST_L may also be asserted when the chip is fully powered, and thus the hot swap controller can transition to state 0 from any other state. When in state 0, all PCI signals are tristated. If local card signals are powered, they also are tristated as the rest of the board may not be powered.

In state 0, all hot swap register bits are in their deasserted, default state (event bits INS and EXT are clear, LOO and EIM are initialized to 0). However, hardware asserts the signal HS_LED, which causes the LED to turn on.

The hot swap controller transitions from State 0 to State 1 when all of the following conditions are met:

- Local reset (LRST_L) is deasserted
- Serial preload is complete
- If ENUM_L is asserted remotely (a signal event frame is generated), either the Skip INS Enable bit or the Remote INS Ready bit is 1 (the default value of both bits is 0). ENUM_L is asserted remotely whenever the ENUM_L Input Signal Mask is disabled (0) in the Event Dispatch Control register.

3.10.3.2 State 1 H/W Connected

In State 1, reset and local configuration has been completed but the switch has not yet been closed. If PI = 1, the PIE bit is also set to 1; if PI = 0, PIE remains 0.

On the transition into State 1, if PI = 0, LOO is initialized to 0 but then is under software control. DHE remains read-only as 0. When PI = 0, configuration registers are accessible in this state.

If PI = 1, LOO and DHE are set to 1 by hardware and cannot be written by software in this state. Because LOO = 1, the LED is on. Device hiding is unconditionally invoked because the switch is open in this state and the configuration registers cannot be accessed from the PCI bus.

After the switch is closed (LSTAT = 0), if Skip INS Enable is 0, the hot swap controller transitions to State 2 to begin the software insertion process. If the switch is closed prior to the transition from State 0 to State 1, then the hot swap controller transitions from State 1 to State 2 on the next clock cycle.

If the switch closes (LSTAT = 0) and Skip INS Enable is 1, then the insertion state is skipped and the hot swap controller transitions directly to State 3 (Installed). If PI = 1, LOO and DHE are cleared on this transition. If the switch is closed prior to the transition from State 0 to State 1, then the hot swap controller transitions from State 1 to State 3 on the next clock cycle.

3.10.3.3 State 2 INS ENUM_L

The hot swap controller enters State 2 when an insertion event is to be signaled. Signaling of this event causes system software to install and initialize software.

There are three cases where the SG2010 can enter this state. The first is when the board has been inserted and the SG2010 has completed reset and the switch has closed. In this case, the hot swap controller has transitioned to state 2 from state 1.

The other two cases occurs after a removal was signaled (the switch was opened), but instead of removing the card the operator closes the handle again, leaving the card inserted. Because the removal has already been signaled, the system software has likely removed related drivers and they must be reinstalled. In this case, the hot swap controller transitions from state 5 to state 2, or from state 5b to state 2.

When transitioning into state 2, the hardware initializes the INS status bit to 1. If EIM is 0, an ENUM_L event is signaled. If PI=1, PIE is 1 in this state; if PI = 0, PIE remains 0.

If PI = 1, LOO and DHE are initialized to 0 on the transition to state 2, and are then controllable by software.

If PI = 0, LOO is not cleared on the transition, but is software controllable. DHE remains read-only as 0.

The LOO bit controls the LED.

The hot swap controller stays in State 2 until the INS status bit is cleared. When INS is cleared, the insertion process completes and the hot swap controller transitions to State 3.

3.10.3.4 State 3 Installed

The hot swap controller transitions to state 3 from state 2 when the INS bit is cleared. This indicates that the card is fully initialized and is under normal operation. If the Skip INS Enable bit was set in State 1, then the hot swap controller transitions directly to State 3 from State 1.

In this state, the LED is fully under software control via the LOO bit. Because the card is fully installed, the PIE bit is set to 0 by hardware. DHE is controllable by software if PI is 1, otherwise it is read-only as 0.

The hot swap controller remains in State 3 until either a software initiated extraction or an operator initiated extraction is indicated. An operator initiated extraction occurs when the switch opens (LSTAT = 1), and either LOO = 0 (no software extraction indicated) or DHE = 0 or PI = 0 (software extraction not supported). When an operator-initiated extraction occurs, the hot swap controlled transitions to State 4.

When a software initiated extraction occurs, it is not necessary to signal ENUM_L because the card has been quiesced. A software extraction is indicated when the switch opens (LSTAT = 1), LOO = 1 to indicate that this is a software extraction, DHE = 1, and PI = 1 to indicate that software extractions are supported. In this case, the hot swap controller transitions directly to State 5b (Hidden) and device hiding goes into effect.

3.10.3.5 State 4 EXT ENUM_L

The hot swap controller enters this state on an operator-initiated extraction. The SG2010 sets the EXT status bit and, if the EIM mask is clear, asserts ENUM_L. The LED remains under software control via the LOO bit. DHE is controllable by software if PI = 1; otherwise, it is read-only as 0. If DHE = 1 and LOO = 1, device hiding is in effect. Because a pending extraction is occurring, PIE = 1 if PI = 1.

The hot swap controller remains in state 4 until software clears the EXT bit. The clearing of the EXT bit causes ENUM_L to deassert and the hot swap controller transitions to the extracted state, State 5.

3.10.3.6 State 5 Extracted

The hot swap controller enters state 5 from state 4 when the EXT status bit is cleared, indicating that the extraction sequence is complete and an insertion is not pending. Software clears EXT after all activity to the card has been quiesced and drivers have been removed.

When in state 5, both the EXT and INS status bits are clear, the ENUM_L interrupt bit is tristated, and the LED is under software control via the LOO bit. If PI = 1, the PIE bit = 1, and DHE is controllable by software; otherwise, DHE is read-only as 0. If DHE = 1 and LOO = 1, device hiding is in effect.

The hot swap controller stays in this state as long as the switch remains open (eventually power is removed as the card is pulled out).

If the switch closes and LSTAT goes to 0, the hot swap controller transitions to state 2. This means that the operator has changed his mind and instead of pulling the card out has closed the handle, leaving the card inserted. In this case the software needs to be reinstalled so the hot swap controller initiates the insertion sequence by transitioning to state 2 and setting the INS bit.

If PI is 1, and DHE and LOO are set to 1 while the switch is still unlocked (LSTAT = 1), then the hot swap controller transitions into State 5b, the hidden state.

3.10.3.7 State 5b Hidden

Transition into State 5b can only happen when PI = 1. The hot swap controller enters State 5b from State 5 when DHE and LOO are both 1 and the switch is unlocked. This activates device hiding on extraction. Device hiding is cancelled only if the switch closes and the insertion process is initiated. The hot swap controller can also transition into State 5b from State 3 on a software-initiated extraction, using the same conditions (DHE, LOO, LSTAT, PI all 1).

When in State 5b, register state is not accessible from the PCI bus. However, both the EXT and INS status bits are clear, the ENUM_L interrupt bit is tristated, and the LED is controlled by the LOO bit. The PIE bit is 1.

The hot swap controller stays in this state as long as the switch remains open (eventually power is removed as the card is pulled out).

If the switch closes and LSTAT goes to 0, the hot swap controller transitions to State 2. This means that the operator has changed his mind and instead of pulling the card out has closed the handle, leaving the card inserted. In this case the software needs to be re-installed so the hot swap controller initiates the insertion sequence by transitioning to State 2 and setting the INS bit.

3.10.4 Device Hiding

The SG2010 implements device hiding to prevent its PCI target state machine from responding when the card is being inserted and/or removed. Device hiding is enabled when PI is 1 and the hot swap controller is in State 0, State 1, or State 5b. When device hiding goes into affect, the SG2010 completes a PCI transaction in progress as early as possible, but does not initiate nor respond to any subsequent transactions on the PCI bus.

The SG2010 also uses the BDSEL_L input pin to control PCI response. BDSEL_L is the first connector pin to break when a card is removed. The SG2010 uses this signal to prevent it from responding to a PCI transaction when the card is physically being removed. When BDSEL_L is detected low and PI is 1, the SG2010 completes a PCI transaction as early as possible if it is in progress, but then does not respond to or initiate any other PCI transactions. The SG2010 will not respond until and unless the card is seated and BDSEL_L is detected high.

Device hiding does not occur if PI is 0.

3.11 Software Generated Frames

The software generated frame (SGF) function allows the SG2010 to generate any type of frame through software control, and send it into the fabric. The SGF registers, described in Section 4.6.9, are part of the CSRs, and are mapped in memory, I/O, and Channel 255 address space. SGFs can be initiated from either the PCI interface or the link interface. The following registers are used:

- SGF Frame – a 144-byte register that contains the frame header and data payload. Up to nine-line frames are supported.
- SGF Destination Address – specifies the PCI memory address of the location where a read completion or write acknowledge frame is written in response to a SGF.
- SGF Bytes Received – indicates the number of read completion or write acknowledge bytes written to memory.
- SGF Control and Status – bits to initiate and check the status of an SGF.

If the SGFs to be generated cause response frames to be returned, local PCI memory space must be allocated for the response frames and the SGF Destination Address must first be initialized to the corresponding PCI memory address. The SGF Destination address range is an aligned 16KB region. The SGF Destination Address register is required to be initialized once before the first response frame is written. The SG2010 automatically moves the address to the next line (16-byte) boundary at the start of every response frame, and wraps the address pointer at the 16KB boundary. If the SGF Write Ack Disable bit is set in the SGF Control register, write acknowledge and bandwidth response frames received in response to an SGF are not written to PCI memory, and in this case memory need not be allocated for these responses.

When a processor is sending an SGF, it must write the frame header and data payload to the SGF Frame register. Software must ensure that a valid frame format and consistent frame field values are used; the SG2010 does not check or fix bad SGF header formats, nor does it perform a chip operation based on the type of frame that is generated. The header of the frame is included in the first line of the SGF Frame register. Software must use a transaction number of 62 (3Eh) if the SGF causes a response frame to be returned. The SG2010 generates the link overhead (sequence number, CRC and line credit byte). The SG2010 positions the link overhead internally or externally based on the value of the Link Overhead bit in the frame, which is written by software in the Frame register. The SG2010 checks the appropriate header fields written to the SGF Frame register to determine the size of the frame. Software must also insure that all unused turns in the path header field are written to 0.

After the SGF Frame register and, if necessary, the SGF Destination Address register are initialized, then the processor writes the SGF Control and Status register to select the output link and send the frame. The Output Link bit is written to 0 to select link 0 and to 1 to select link 1. The output information is link-based and not port-based. This is necessary for special line credit update frames, which are performed on a link-basis. Either the Link State table or the Port State table may be used to obtain a link-to-port mapping.

When the Send_SGF bit is written with a 1, the SG2010 sends the information contained in the SGF Frame register to the output buffer of the selected link. When the frame is sent or discarded, the SG2010 clears the SGF Bytes Received register.

The SG2010 clears the Send_SGF bit after the frame is sent or if it is discarded due to an error or link down condition. If the frame is discarded, the SGF Not Sent bit is set. The SG2010 clears these bits when the next SGF is sent.

If the SG2010 generates a frame requiring a write acknowledge, read completion, or bandwidth response it sets the response outstanding (RESP_OUT) bit when the frame is sent. The bit remains set until the write acknowledge, bandwidth response, or last read completion frame is received and written. If software determines that an expected response frame is not going to be received, it can write 1 to the RESP_OUT status bit to clear the bit, and to reset the SGF function so that it no longer waits for frames in response to that SGF.

The SG2010 detects an SGF response frame by checking to see if the Request Transaction Number is equal to 62 (3Eh). When a read completion or write acknowledge frame is received in response to an SGF, the SG2010 copies the failure type in the header into the four-bit Completion Status (CMPSTAT) field in the SGF Control and Status register. A failure type of Fh indicates the SGF completed normally. If the response frame is a bandwidth response frame, the secondary operation is stored in the CMPSTAT field. CMPSTAT is not updated until the next response frame is received. Currently, the SG2010 does not take any action based on the failure type (that is, re-send on a lockout, and so on). Software must check the completion status and take appropriate action based on the status.

When the SG2010 detects an incoming response frame in response to an SGF, it writes the frame to PCI memory using the Destination Address Pointer as the PCI address. In the case of a write acknowledge or bandwidth response, the SGF Write Ack Disable bit must be clear to allow the write to memory. As the SG2010 writes the response frame to memory, it increments the SGF Byte Count by four for every Dword it writes. After the write to PCI memory is complete, the SG2010 updates the address to the next 16-byte aligned value, and wraps at a 16KB boundary. In other words, the SG2010 resets Destination Address bits [3:2] to 0, and increments bits [13:4].

If the SG2010 reaches the 16KB boundary in the middle of writing a response on the PCI bus, the SG2010 master terminates at the 16KB boundary and resumes the write at the beginning of the buffer.

The read response to an SGF may consist of several completion frames. The SG2010 detects the last completion frame by checking the Decomposition Type header bit.

After the last read completion is written to memory or when a write acknowledge or bandwidth response frame is received, the SG2010 clears the RESP_OUT bit and sets the SGF_DONE state bit in the SGF Control and Status register and also sets the SGF_Done event bit. If no response frame is required, then the SG2010 signals SGF_Done as soon as the frame is sent (or discarded due to link down). The SGF_DONE register state bit is cleared either when software clears the SGF_Done event bit in the Raw Event Status register or when the next SGF is sent by writing 1 to the SEND_SGF bit.

The SEND_SGF, RESP_OUT, and SGF_DONE bits create a state table for the status of the SGF, as shown in Table 3–38.

Table 3–38 SGF State Table

State	SEND_SGF	SGF Not Sent	RESP_OUT	SGF_DONE
0: Idle	0	0	0	0
1: SGF pending, not sent	1	0	0	0
2a: SGF frame discarded	0	1	0	0
2b: SGF sent, awaiting response	0	0	1	0
2c: SGF sent, no response required	0	0	0	1
3: SGF response frame received	0	0	0	1

Using the states in Table 3–38, the state transition for an SGF not requiring a completion or a response is simply 0→1→2c→0. For an SGF requiring a completion or a response, the state transition is 0→1→2b→3→0. If an SGF is discarded before it is sent, the state transition is 0→1→2a→0.

Generation of any type of response frame through the SGF mechanism is not recommended since these frames should be generated in hardware in response to a write or read request. Reception of unexpected response frames may cause unpredictable results at the terminus of the frame. The SG2010 does not allow the generation of a bandwidth response frame - if that type of frame is detected the SG2010 does not send the frame.

3.12 Software Generated Transactions

Software generated transactions (SGTs) allow a processor to initiate a configuration or I/O transaction on a remote PCI bus through software control. The SG2010 implements a set of registers, the SGT Configuration Address and SGT Configuration Data registers, to initiate a configuration transaction and a set of registers, the SGT I/O Address and SGT I/O Data registers, to initiate an I/O transaction. These registers are located in the Gateway configuration register set, and are described in Section 4.8.4.

The SG2010 initiates a software generated transaction only when the registers are accessed from the link interface. The SG2010 does not support software transaction generation when the registers are accessed from its PCI bus. The SG2010 does not check the outgoing configuration or I/O transaction, nor does it perform any chip operations based on the transaction – it just initiates it on the PCI bus. The SG2010 does not respond to any software generated transactions that it initiates.

To generate a transaction, the corresponding SGT Enable must be set in the Gateway Chip Control configuration register. Accessing the SGT Configuration Data or SGT I/O Data register causes the SG2010 to initiate a transaction on its PCI bus. Before the data register is accessed, the SGT Configuration or I/O address must first be set up. In both cases, the address is used on the PCI bus exactly as written in the register. When the address register is written from the link and the SGT Enable bit is set, the SG2010 sets the Configuration or I/O SGT Busy bit in the Gateway Chip Status configuration register.

The SG2010 performs a configuration write when the SGT Configuration Data register is written, using the same data that was used for the register access. The byte enables used for the PCI transaction correspond to the byte mask used for the SGT Configuration Data access. Thus, in order to perform a byte access, the appropriate byte mask must be used when writing or reading the SGT Configuration Data register.

Similarly, the SG2010 initiates an I/O write when the SGT I/O Data register is written. If a response frame is required, the SG2010 generates a write acknowledge frame after the transaction completes with either a TRDY_L, target abort, master abort, or retry time-out. The SG2010 supports only a single Dword write when performing software generated transactions. The incoming write frame must be a single Dword write.

The SG2010 initiates a configuration read when the SGT Configuration Data register is read, or an I/O read when the SGT I/O Data register is read. Again, the byte enables used for the PCI transaction correspond to the byte mask used for the data register read request. The SG2010 generates a read completion frame after the transaction completes and data is returned with TRDY_L, or a target abort, master abort, or retry time-out is detected. The SG2010 supports only a single Dword read when performing software generated transactions.

When the software generated transaction completes, the SG2010 clears the Configuration or I/O SGT Busy bit in the Gateway Chip Status configuration register.

Error handling for PCI termination and parity errors is performed in the same way as for other link-to-PCI transactions.

If the data register is written and the corresponding SGT Enable bit is not set, the SG2010 discards the data and sends back a normal write acknowledge frame. If the data register is read and the SGT Enable bit is not set, the SG2010 returns all 0s as data in the read completion frame. Regardless of the state of the SGT Enable bit, if the data registers are accessed from the PCI bus, the SG2010 asserts TRDY_L and either discards the write data or returns all 0s on reads.

3.13 Semaphores

The SG2010 implements eight 8-bit semaphores that can be used by software to reserve resources. The semaphores are not tied in hardware to specific resources; this is managed in software. All software using the semaphores must have a common understanding of the semaphores’ purpose.

Each semaphore consists of eight Dword-aligned registers, one register per semaphore operation. A semaphore is altered atomically when a read operation is performed to one of its registers. When a semaphore register is read, the current value of the semaphore is returned, and then the operation associated with that register is performed.

The following semaphore operations are supported:

- Clear – the semaphore is cleared to 00h
- Set – the semaphore is set to 01h
- Decrement – the semaphore is decremented by 1
- Increment – the semaphore is incremented by 1
- Increment if 0 – the semaphore is incremented if the current value is 0
- Increment if not 0 - the semaphore is incremented if the current value is not 0
- Two reserved operations – no semaphore operation is performed

The current value is the value that is returned in the read, and is the value before the semaphore operation. Writes to the semaphore registers are discarded and have no effect. The semaphores are sticky for decrements at 0, and for increments at FFh; that is, they do not wrap. If a semaphore Dword location is read and all of the byte enables for that Dword are disabled, the semaphore operation is not performed.

3.13.1 Semaphore RMW Frames

The SG2010 performs a semaphore operation when any PCI register read, address-routed read request, or path-routed read request accesses a semaphore register. However, the SG2010 also supports access of the semaphore registers through a Channel 255 path-routed read request frame that specifies the read-modify-write operation. When the SG2010 receives such a frame, it extracts the eight-bit semaphore number and the three-bit semaphore operation from the header. The SG2010 uses the semaphore number and the semaphore operation number to construct the semaphore CSR offset. The SG2010 then reads the appropriate semaphore register, returns the current value in a read completion frame, and performs the semaphore operation.

If an unsupported semaphore number is used (any number except 0-7) then a read completion frame with a Range failure type is returned. If a Channel Number other than 255 is specified, a Channel Inactive failure type is returned.

3.14 General Purpose I/O Interface

The SG2010 general purpose I/O (GPIO) interface allows software control of a signal pin interface, or can be used to provide hardware state information to software. The SG2010 implements four dedicated GPIO[3:0] signal pins, with an additional four, GPIO[7:4], that can be supported through reallocation of arbiter pins REQ_L[8:7] and GNT_L[8:7]. These arbiter pins are assigned to the GPIO function when the Pin Mode bit is set in the Arbiter Control register in Gateway configuration space.

The GPIO pins are controlled through a GPIO register interface in the Gateway configuration registers. The register interface controls both the direction of each GPIO pin (input only or bidirectional) as well as the value on the GPIO signal pin.

The GPIO Set Direction and GPIO Clear Direction registers are used to configure a GPIO pin to be an input-only or a bidirectional pin. Each bit in the GPIO direction registers corresponds to a GPIO signal. When a GPIO direction bit is 1, the corresponding GPIO pin is configured to be bidirectional and the SG2010 drives the value of the corresponding bit in the GPIO Data register onto the signal. When a GPIO direction bit is 0, the corresponding GPIO pin is configured to be input-only and the SG2010 can only sample the value of the GPIO signal. Writing a 1 to a GPIO Set Direction bit sets the bit to 1, assigning the signal as bidirectional. However, if GPIO[7:4] pins are assigned to the arbiter, the direction of these bits is always input only (0), regardless of what is written. Writing a 1 to a GPIO Clear Direction bit clears the bit to 0, assigning the signal as an input. If a 1 is written to the same bit in both registers with the same write operation, the set operation is dominant over the clear operation. Writing a 0 to either register has no effect. Reading either register returns the current value of the GPIO Direction register.

The GPIO Set Data and GPIO Clear Data registers are used to assign a value to a GPIO pin that is configured to be bidirectional. Each bit in the GPIO data registers corresponds to a GPIO signal. When the GPIO Set Data register is read, it returns the current value on the GPIO signal pins, regardless of the direction of the pin. When the GPIO Clear Data register is read, it returns the value written to the GPIO Data register.

When a GPIO data bit is 1 and the corresponding GPIO direction bit is 1, the GPIO pin is driven high. When a GPIO data bit is 0 and the corresponding GPIO direction bit is 1, the GPIO pin is driven low.

Writing 1 to a GPIO Set Data bit sets the corresponding GPIO Data bit. Writing 1 to a GPIO Clear Data bit clears the corresponding GPIO Data bit. When 1 is written to the same bit in both registers with the same write operation, the set operation dominates the clear operation. Writing 0 to either register has no effect.

3.15 Doorbell Interface

The SG2010 implements 32 doorbells that enable the assertion of a local INTA_L interrupt. Each of the 32 doorbells is assigned a doorbell interrupt bit and a doorbell interrupt mask bit. The Doorbell registers controlling this function are mapped into CSRs. They are considered to be a device-specific feature and not a part of the event handling logic. They neither use the EMU event counters nor do they generate event messages.

The Set IRQ and Clear IRQ registers contain the doorbell interrupt bits. When 1 is written to a bit in the Set IRQ register, the doorbell bit is set and, if the corresponding mask bit is clear, INTA_L is asserted. When 1 is written to a bit in the Clear IRQ register, the doorbell bit is cleared. If the corresponding mask bit is clear, INTA_L is conditionally deasserted (it can be asserted for other reasons). A read of either the Set IRQ or Clear IRQ registers returns the current state of the doorbell interrupts.

The Set IRQ Mask and Clear IRQ Mask registers contain the doorbell interrupt mask bits. When 1 is written to a bit in the Set IRQ Mask register, the corresponding mask bit is set. When 1 is written to a bit in the Clear IRQ Mask register, the corresponding mask bit is cleared. A read of either the Set IRQ Mask or Clear IRQ Mask registers returns the current state of the doorbell mask.

The assertion and deassertion of the INTA_L signal may depend on other events, since it is shared with the event handler.

3.16 Scratchpad Registers

The SG2010 implements eight 32-bit read/write registers for use as scratchpad registers. These registers have no side effects and do not have any effect on any SG2010 functions.

3.17 PCI Arbiter

The SG2010 implements a PCI bus arbiter that supports nine bus masters plus SG2010's bus master. The arbiter uses a two-level, rotating priority algorithm. The arbiter may be enabled or disabled through a strapping pin. If the arbiter is disabled, an external PCI bus arbiter must be used.

3.17.1 PCI Arbiter Signal Interface

Table 3–39 describes the PCI Arbiter signal pins.

Table 3–39 PCI Arbiter Signal Pins

Pin Name	I/O	Description
REQ_L[0]/AGNT_L	I	When the arbiter is enabled, this pin is used as a request input from an external PCI bus master. When the arbiter is disabled, this pin is used as the grant signal from the external arbiter to SG2010's PCI bus master logic.
GNT_L[0]/AREQ_L	O	When the arbiter is enabled, this pin is used as a grant output to an external PCI bus master. When the arbiter is disabled, this pin is used as the request signal to the external arbiter from SG2010's PCI bus master logic.
REQ_L[8:1]	I	When the arbiter is enabled, these pins may be used as request inputs from external PCI bus masters. Unused request inputs[8:7] may be used as GPIO pins by setting the Pin Mode bit in the Arbiter Control register.
GNT_L[8:1]	O	When the arbiter is enabled, these pins may be used as grant outputs to external PCI bus masters. Unused grant outputs[8:7] may be used as GPIO pins by setting the Pin Mode bit in the Arbiter Control register.

If the arbiter is enabled, SG2010's PCI request and grant signal connections are internal to the chip.

3.17.2 PCI Arbiter Operation

3.17.2.1 PCI Arbitration Algorithm

The PCI arbiter uses a two-level rotating priority algorithm. Each bus master is a member of either the low priority group or the high priority group. Bus masters using request lines [8:2] occupy only one arbitration slot in a priority group. The SG2010 and bus masters using request lines [1:0] may occupy one or two arbitration slots if they are in the high priority group and one slot if they are in the low priority group. The group in which a bus master resides and the number of slots it occupies are configured using the Arbiter Control register in Gateway configuration space. After reset, the SG2010 occupies one arbitration slot in the high priority group and the remaining bus masters are placed in the low priority group.

Priority rotates evenly among each arbitration slot in the high priority group and between each arbitration slot in the low priority group. The low priority group occupies a single slot in the high priority group.

An arbitration time period starts after the assertion of FRAME_L on the PCI bus and continues until the next assertion of FRAME_L. Priorities for a given arbitration time period are assigned after the beginning of each PCI transaction. The arbitration slot that received the grant for the previous PCI transaction becomes the lowest priority arbitration slot during the next arbitration time period.

During any arbitration time period, GNT_L is asserted to the highest priority bus master that has its request asserted. The grant signal may be deasserted at any time if a higher priority bus master subsequently asserts its request signal. If no bus master has a request asserted then the bus remains parked (grant is asserted) at either the last bus master that used the bus or at the SG2010.

If the arbiter detects that a bus master has its grant asserted with the PCI bus idle for 16 consecutive cycles then the arbiter will deassert that bus master's grant signal and move that master to the lowest priority position, and the grant may be asserted to another bus master. If no other bus master has its request asserted then the bus is parked at the SG2010. In order for that bus master to be serviced again, the bus master must deassert its request signal for one or more clock cycles.

3.17.2.2 Bus Parking

The PCI bus is parked (driven) when it is idle by the bus master that has a grant signal asserted. When SG2010's PCI arbiter is enabled, the SG2010 may be programmed to always park the bus itself or to park the bus at the last master that used the bus. Bus parking is selectable by a bit in the Arbiter Control register.

Only AD[31:0], CBE_L[3:0] and PAR are bus parked. The 64-bit extension signals (AD[63:32], CBE_L[7:4], PAR64) are not bus parked as they are pulled up through external resistors.

3.17.2.3 Disabling the Arbiter

The PCI bus arbiter may be disabled and an external arbiter used instead. The arbiter is disabled when the PR_AD[6] signal is sampled low on the deasserting edge of reset. When the arbiter is disabled, the REQ_L[0] output is then used as the AGNT_L input for the SG2010, and the GNT_L[0] output is used as the AREQ_L output for the SG2010. The SG2010 arbiter logic ignores the remaining reset inputs and drives high the remaining grant outputs, except for those pins that are reconfigured as GPIO signals. Unused arbiter inputs should be pulled high through an external resistor.

3.18 ROM Interfaces

The SG2010 implements a pin interface that is shared between the serial ROM and parallel ROM.

3.18.1 ROM Programming Interface

The serial and parallel ROMs are programmed through the SG2010 ROM register interface, located in the Gateway configuration registers. The SG2010 ROM register interface comprises the following 32-bit registers:

- ROM Data register
- ROM Address register, consisting of
  - 24 bits of ROM Address
  - Serial/Parallel ROM Operation Select bit
  - Start/Busy Control bit
  - Parallel ROM Byte Operation Select
- ROM Setup register for the parallel ROM interface, consisting of
  - Multiple Device Mode enable bit
  - 2-bit Access Time Control
  - 2-bit Address Pipeline Control
  - 8-bit read/write strobe mask

The ROM control registers are described in Section 4.8.5.

3.18.1.1 Writing through the ROM Register Interface

The basic programming interface for writing to the parallel and serial ROM is the same. The register interface supports an aligned four-byte write to either ROM, or a single byte write to the parallel ROM. In order to perform a four-byte write, four bytes of data are first written to the ROM Data register. A second write to the ROM Address register writes the Dword-aligned ROM address, the Parallel/Serial ROM Select bit to 0 for an SROM write or 1 for a parallel ROM write, and the Start/Busy flag to 1 to select and start the write operation. If a parallel ROM operation is specified, the Parallel ROM Byte Operation Select bit should be written with 0.

If a parallel ROM byte write operation is specified, the data should be written to the low eight bits of the ROM Data register. When the Start/Busy flag is written, the Parallel ROM Byte Operation Select bit should be written with 1.

When the write operation is complete and the ROM interface is ready for another operation, the SG2010 clears the Start/Busy flag to 0.

3.18.1.2 Reading through the ROM Register Interface

The basic programming interface for reading to the parallel and serial ROM is the same. The register interface supports an aligned four-byte read from either ROM, or a single byte read from the parallel ROM. To initiate the read, the ROM Address register writes the Dword-aligned ROM address, the Parallel/Serial ROM Select bit to 0 for an SROM read or 1 for a parallel ROM read, and the Start/Busy flag to 0 to select and start the read operation. If a parallel ROM operation is specified, the Parallel ROM Byte Operation Select bit should be written with 0.

When the read operation is complete and the ROM interface is ready for another operation, the SG2010 sets the Start/Busy flag to 1. The read data can then be read from the ROM Data register. If only a byte of data is read, it is located in the low eight bits of the ROM Data register.

3.18.2 Vital Product Data (VPD)

The SG2010 supports the Vital Product Data (VPD) enhanced capability function. This function allows a processor to access product information through a standard configuration register interface. The VPD configuration registers control the serial ROM interface. VPD read and write accesses are always four bytes, Dword-aligned.

The VPD registers are:

- The 32-bit VPD Data register
- The 16-bit VPD Address register
  - A 7-bit VPD address field
  - A Start/Busy flag

The VPD interface accesses only that portion of the serial ROM designated to be VPD space. The SG2010 defines a 256-byte VPD starting at serial ROM offset 0. VPD space is partitioned further into read-only and read/write space. The read only portion consumes the first 128 bytes of VPD space. The read/write portion consumes the remaining 128 bytes of VPD space. VPD read-only space cannot be written from the VPD register interface.

The SG2010 ROM programming interface must be used to write VPD read-only space, or to access a serial ROM location outside of VPD space. The SG2010 ROM programming interface can read or write any serial ROM location.

The VPD read and write operations are similar to the SROM programming operations described in Section 3.18.1. To read VPD, the VPD Address register is written with the Dword-aligned VPD address offset and a VPD Start/Busy flag value of 0. Writing the Start/Busy bit with a 0 causes the SG2010 to initiate the VPD read. The SG2010 appends the VPD offset onto the VPD base address, which is 0. When the Serial ROM returns the read data, the SG2010 places it in the VPD Data register and sets the Start/Busy flag to 1.

To write VPD, the VPD write data is first written to the VPD Data register. The VPD Address register is then written with the Dword-aligned VPD address offset and a VPD Start/Busy flag value of 1. Writing 1 to the Start/Busy bit causes the SG2010 to initiate the VPD write. When the serial ROM write is completed, the SG2010 clears the Start/Busy flag to 0. If a write is attempted to the read-only VPD space, the SG2010 does not perform a serial ROM operation but immediately clears the Start/Busy flag to 0.

3.18.3 Serial ROM Interface

The Serial ROM stores register preload write data and vital product data (VPD) information. The first 256 bytes of the serial ROM are reserved for use by VPD, the remaining serial ROM locations, starting at byte offset 256 (100h) are used for register preload.

The SROM interface supports serial peripheral interface (SPI) compatible SROMs. The serial data format uses an eight-bit OPCODE and a 16-bit address. SROMs compatible with this format include the Microchip 25AA160/25LC160/25C160 family.

The SROM pin interface consists of the following signals:

- SR_DI – serial data ROM input
- SR_DO – serial data ROM output
- SR_CS_L – serial ROM chip select
- SR_CK – serial ROM clock

Signal pins SR_DI, SR_DO, and SR_CK are shared with the parallel ROM interface pins. Signal pin SR_CS_L is a dedicated pin. When the SG2010 asserts SR_CS_L low, the SG2010 is selecting the remaining three signal pins for the serial ROM function.

The register interface specifies only whether a write or a read operation is to be performed. The SG2010 always performs a write enable before every SROM write. The SROM performs its own write disable operation after each write.

3.18.3.1 Serial Address and Data Organization

The ordering of bits sent to and received from the SROM is from most significant bit to least significant bit for both address and data. For the 16 bit address, the specific *bit* sequence is AD15, AD14, AD13 … AD0. For each byte of data, the specific sequence is D7, D6, D5 … D0.

When multiple bytes are read from the SROM, the order of bytes is least significant to most significant. For a four-byte operation, the specific byte sequence is Byte 0, Byte 1, Byte 2 and Byte 3. The specific 32-bit sequence for such an operation is as follows:

Byte0-D7, Byte0-D6, Byte0-D5,… Byte0-D0, Byte1-D7,… Byte3-D0.

The SG2010 supports four-byte writes and four-byte reads.

3.18.3.2 Serial ROM Write Operation

Figure 3–21 shows the four-byte serial ROM sequence.

The SG2010 executes a Write Enable (WREN) instruction prior to each SROM Write Operation to enable the SROM Write latch. The SROM sequencer deasserts the SR_CS_L signal when the WREN instruction completes. The WREN operation is described in Section 3.18.3.4.

The write operation consists of the following steps

1. The SG2010 asserts SR_CS_L and shifts the instruction, address and data sequence to the SROM as shown in Figure 3–21. This causes the serial ROM to:
   a. Write the 16-bit address into the two-byte address field in its internal CSR.[8]
   b. Write Data into the D0, D1, D2 and D3 bytes in its CSR.

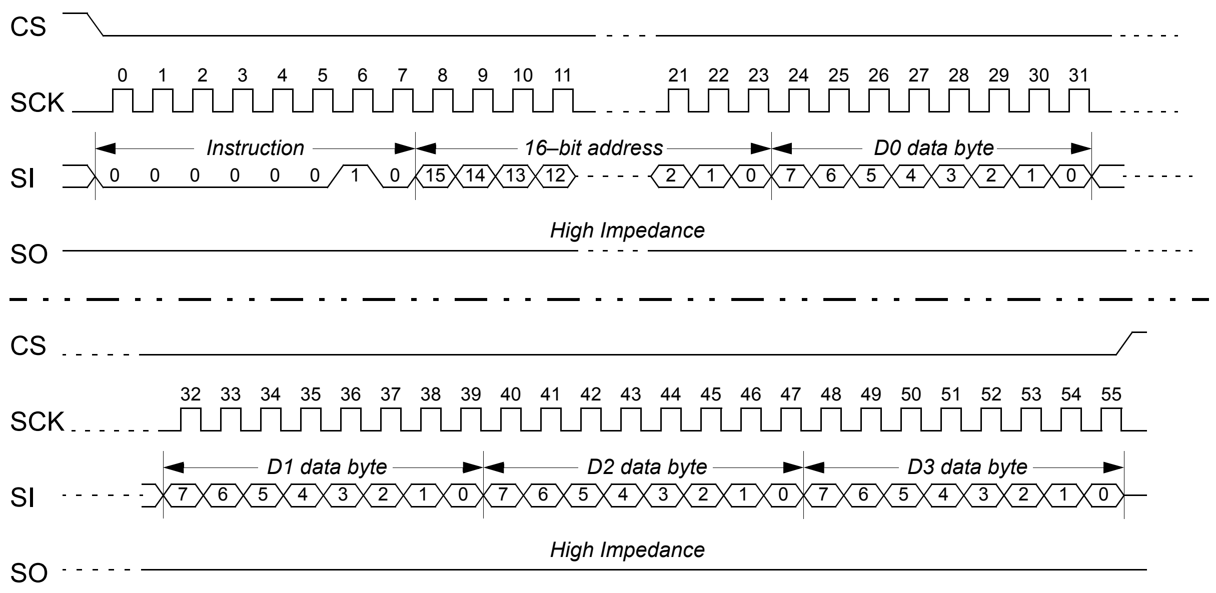
c. Write OPCODE 0000.0010h into the OPCODE byte in its CSR.

d. Write 1 to the Busy bit in its CSR status register. This event triggers the write operation internally to the SROM.

2. The SG2010 completes the write operation and deasserts SR_CS_L. The write operation consumes 55 clock cycles.

3. The SG2010 then enters a polling routine using the RDSR (Read Status register) operation described in Section 3.18.3.5.

Figure 3–21 Serial ROM Write Timing Diagram

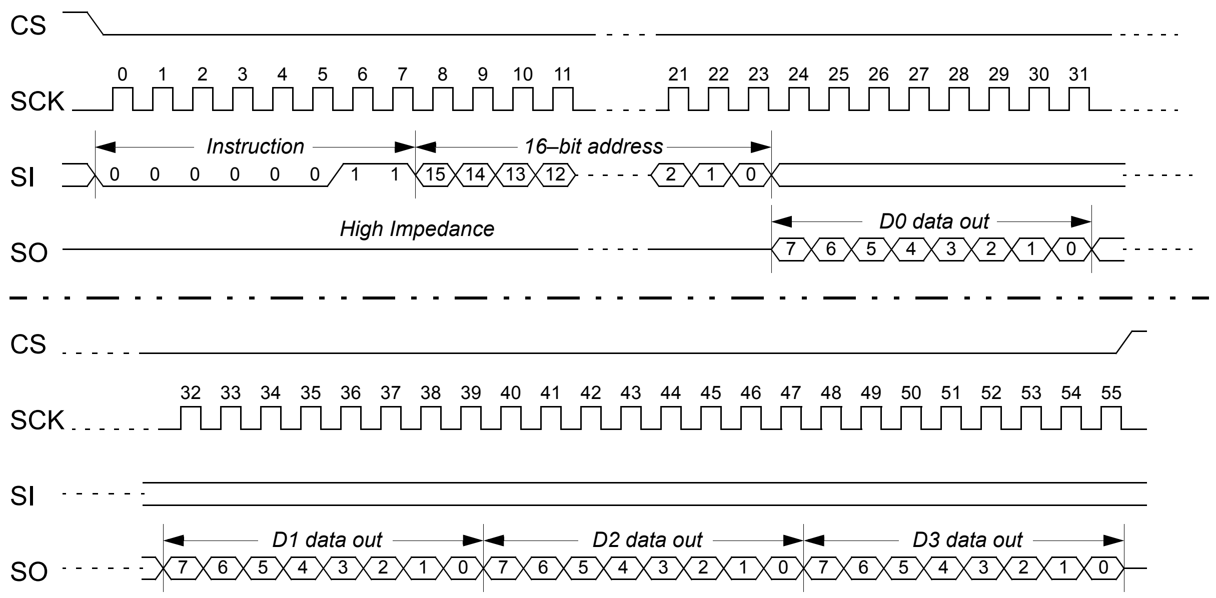


3.18.3.3 Serial ROM Read Operation

Figure 3–22 shows the Serial ROM read timing diagram.

8. A 16-bit address assumes a 64K SROM (8K × 8 bit). For a 16K SROM (2K × 8 bit), only the lower 12 address bits are used. The upper four bits should be written to 0.

Figure 3–22 Serial ROM Read Timing Diagram



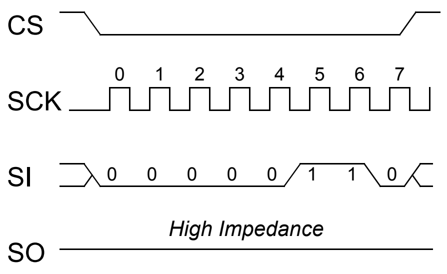
The read operation consists of the following steps:

1. The SG2010 asserts SR_CS_L and shifts the instruction and address sequence to the serial ROM. This causes the serial ROM to
   a. Write the 16-bit address into the two-byte address field in its CSR.
   b. Write OPCODE 0000.0011 into the OPCODE byte in its CSR.
   c. Writes 1 to the Busy bit in the SROM CSR status register. This event triggers the Read operation internally to the SROM.
2. The SG2010 completes the read operation. The read operation consumes 55 clock cycles.
3. The SG2010 receives the serial data and places it in the ROM Data Register.
4. The SG2010 sets the Start/Busy flag to 1 in the ROM Address register to indicate that the read operation is complete.

3.18.3.4 Serial ROM Write Enable Operation

Figure 3–23 shows the Serial ROM write enable timing diagram.

Figure 3–23 Serial ROM Write Enable Timing Diagram



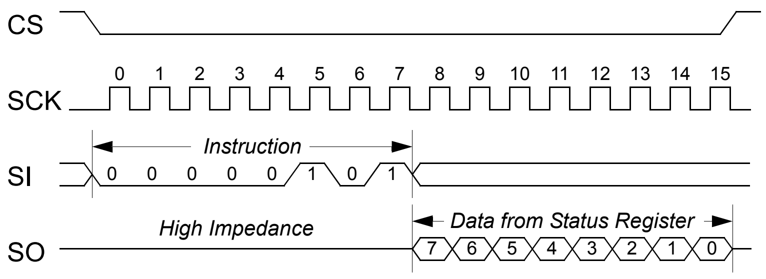
This SROM transaction is used to enable the Write Enable latch internal to the SROM. This latch must be enabled for the SG2010 to perform an SROM write. The SG2010 automatically performs the write enable operation before any SROM write. The following sequence is used for the write enable operation:

1. The SG2010 asserts SR_CS_L and shifts the instruction sequence to the serial ROM. This causes the serial ROM to write OPCODE 0000.0110 into the OPCODE byte in its CSR.
2. The SG2010 deasserts SR_CS_L. The write enable operation consumes eight clock cycles.
3. When the SROM sequencer completes the write enable sequence, it then proceeds with the serial ROM write operation.

3.18.3.5 Serial ROM Read Status Register Operation

Figure 3–24 shows the Serial ROM read status timing diagram.

Figure 3–24 Serial ROM Read Status Timing Diagram



This SROM transaction is used to read the SROM status register. The SG2010 automatically performs the read status operation immediately after any SROM write to determine when the write is complete internally to the SROM. The SG2010 performs the read status operation every 2ms until the status indicates that the write is complete. The following sequence is used for the read status operation:

1. The SG2010 asserts SR_CS_L and shifts the instruction sequence to the serial ROM. This causes the serial ROM to write OPCODE 0000.0101 into the OPCODE byte in its CSR.
2. The serial ROM shifts out the contents of its status register.

3. The SG2010 deasserts SR_CS_L. The read status operation consumes 16 clock cycles.
4. The SG2010 checks bit [0] (BUSY) the status register to determine whether the write has completed. If the write has completed, the SG2010 clears the Start/Busy bit in the ROM Address register to indicate that the serial ROM write operation is complete. If the write has not completed, the SG2010 performs this operation again 2ms later.

3.18.4 Parallel ROM Interface

The parallel ROM can be used to store device-specific initialization code or boot code.

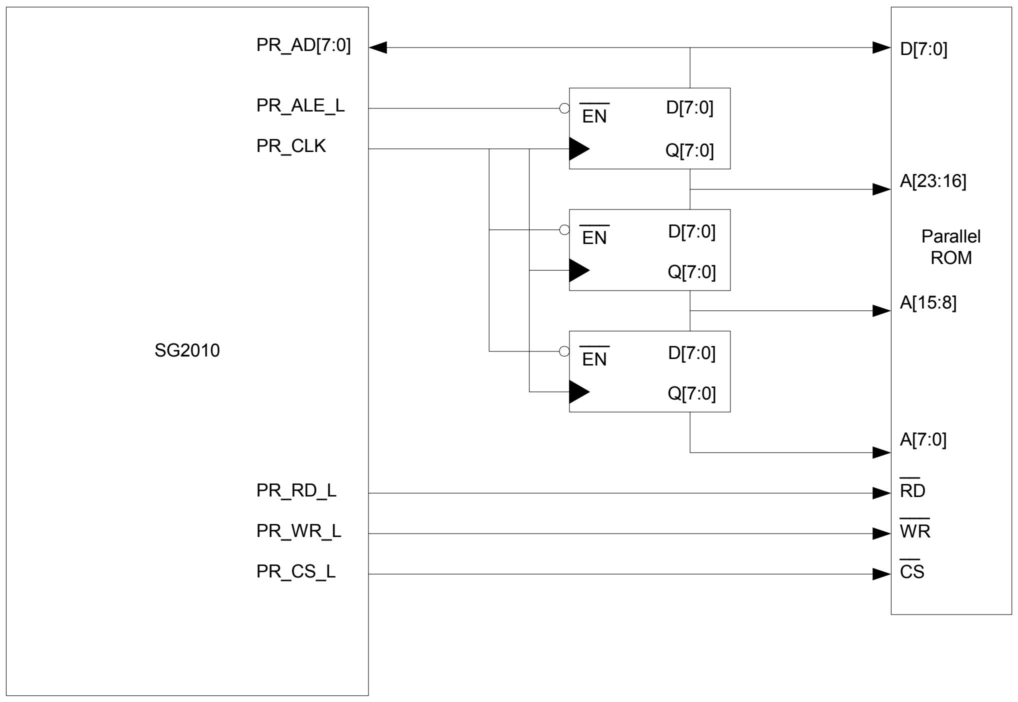
The parallel ROM is read/write accessible through the ROM configuration registers. The parallel ROM can also be mapped for read access only into PCI memory space using the Expansion ROM Base Address register.

The parallel ROM interface has an eight-bit shared address/data bus and control signals, as follows:

- PR_AD[7:0] – shared eight-bit address and data bus
- PR_CS_L – active low parallel ROM chip select
- PR_WR_L – active low write strobe
- PR_RD_L – active low read strobe
- PR_ALE_L – active low address latch control
- PR_CLK – address latch clock

A parallel ROM has separate address and data buses. To interface the parallel ROM to SG2010’s multiplexed ROM bus, the address must be latched externally. The data bus must be eight bits wide. The address bus may be any width up to 24 bits. Parallel ROM signal connections are shown in Figure 3–25.

Figure 3–25 Parallel ROM Connections

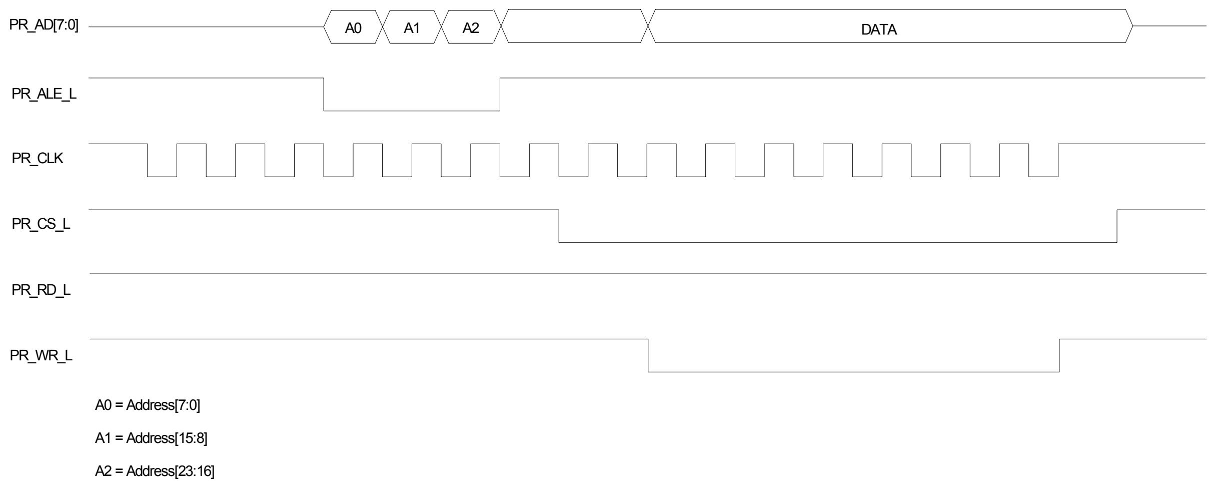


The SG2010 drives the address eight bits at a time in three consecutive PR_CLK clock cycles, starting with the eight least significant bits. If the parallel ROM is accessed through the Expansion ROM BAR, note that the low 24 address bits of the PCI address are driven as the parallel ROM address, regardless of the size of the Expansion ROM BAR range. The SG2010 does not mask upper address bits based on the size of the Expansion ROM BAR range.

On the PR_CLK cycle following the last data phase, the SG2010 drives the parallel ROM control signals. The access time and strobe mask fields in the ROM Setup configuration registers specify the timing of these control signals.

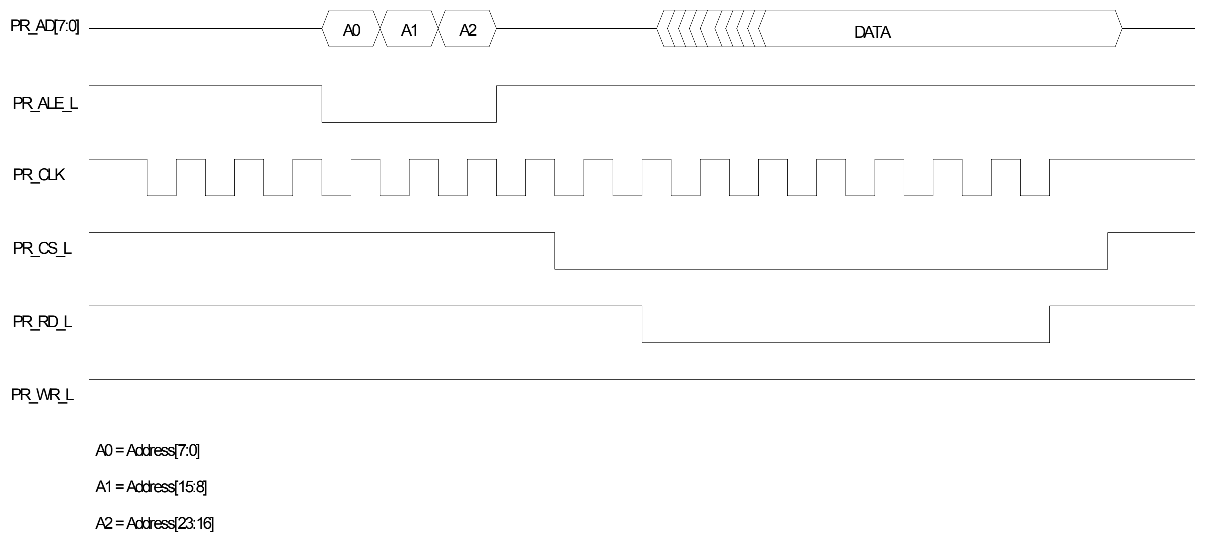
If the operation is a write, then write data is driven one-half PR_CLK cycle earlier than PR_CS_L assertion. Write strobe PR_WR_L is asserted low and deasserted high with the timing specified by the strobe mask. Chip select PR_CS_L remains asserted for the amount of time specified by the access time field. The SG2010 continues to drive the write data until the chip select is driven high (deasserted). Figure 3–26 shows a timing diagram of the SG2010's pins for a parallel ROM write.

Figure 3–26 Parallel ROM Write Timing Diagram



If the operation is a read, chip select PR_CS_L is asserted low a half PR_CLK cycle after the last address phase completes. Read strobe PR_WR_L is asserted low and deasserted high with the timing specified by the strobe mask. Chip select PR_CS_L remains asserted for the amount of time specified by the access time field. The SG2010 latches the parallel ROM read data on the deasserting edge of PR_RD_L. Figure 3–27 shows a timing diagram of the SG2010's pins for a parallel ROM read.

Figure 3–27 Parallel ROM Read Timing Diagram



If a Dword access is specified, the parallel ROM sequencer performs four byte accesses to either write or read the Dword. If a byte access is specified, the parallel ROM sequencer performs only a single byte access.

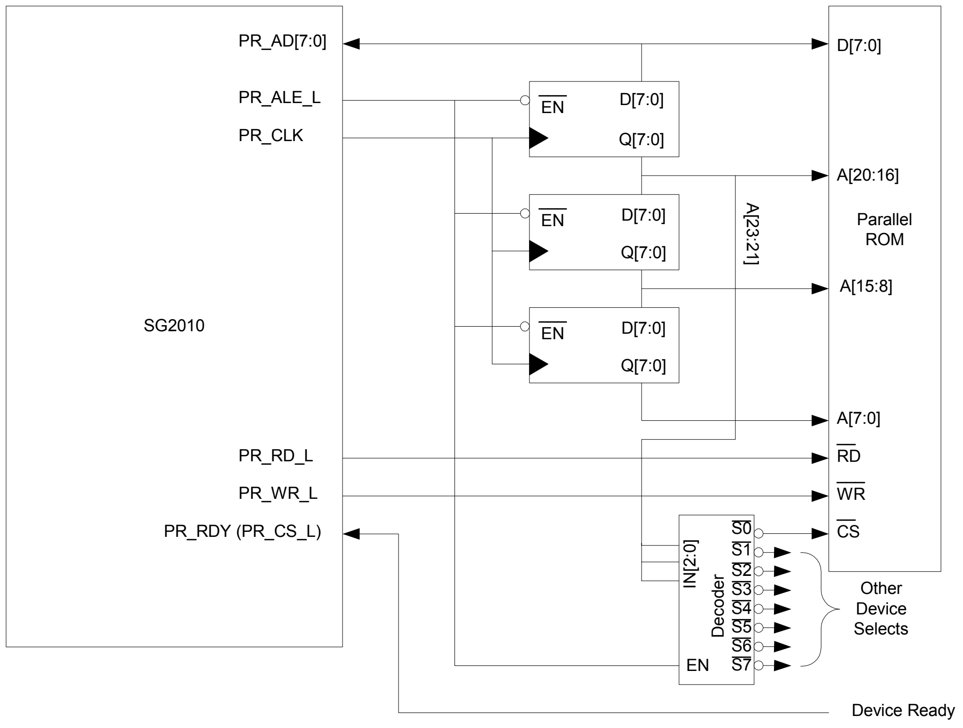
3.18.4.1 Attaching Multiple Devices

The SG2010 parallel ROM interface supports the attachment of multiple slave devices. The Multiple Device Mode bit in the ROM Setup configuration register configures SG2010's parallel ROM interface to operate in this mode. The two major differences in this mode are the requirement for external device select generation, and the capability of extending read and write accesses. Figure 3–28 shows the connections for using multiple device mode.

Instead of using the PR_CS_L signal as a single chip select, upper ROM address bits must be externally decoded to generate chip selects. The number of address bits used and their encodings are application specific. For example, the upper three address bits [23:21] may be decoded externally to provide up to eight device select signals, and the remaining bits [20:0] are used for the address.

The PR_CS_L bit is configured as an input in multiple device mode, and can be used to extend the access time of the read or write. When PR_CS_L is driven low by external logic during a read or write access, the access time is extended by the amount of time that the signal remains low. After the signal is driven high, the SG2010 completes the access using the remaining access and strobe time that existed when PR_CS_L was driven low.

Figure 3–28 Multiple Device Mode for Parallel ROM Interface



3.19 Diagnostic Interfaces

3.19.1 JTAG

The SG2010 is fully compliant with the IEEE 1149.1a-1993 Boundary Scan Specification (known informally as "JTAG"). The SG1010 includes the following pins for IEEE 1149.1 support: TRST_L (Test Reset), TDI (Test Data In), TMS (Test Mode Select), TCK (Test Clock) and TDO (Test Data Out). The first four pins are input-only, while TDO is output-only.

All of the input pins are pulled high internally. TDO is an output-only and drives low when TRST is high. The SG2010's implementation of IEEE 1149.1 includes the required instructions BYPASS, EXTEST, and SAMPLE/PRELOAD, and the optional instructions IDCODE and RUNBIST.

For more information, see the IEEE Standard Test Access Port and Boundary Scan Architecture 1149.1-1990, IEEE Std. 1149.1a-1993, and IEEE Std. 1149.1b-1994.

3.19.2 LED Signal Interface

The SG2010 implements eight signals that can be used for LED control based on link state or software control. There are two sets of four signals, one set per link: LED0_L[3:0] and LED1_L[3:0]. The four signals per link are allocated to each of the four differential pair receivers per link.

These signals are shared with the TESTMUX[9:0] signals. Use of the TESTMUX[9:0] signals automatically overrides use of these respective signals for LED control.

The LED*x* signals are controlled using the LED Control register, located in CSR register space and described in Section 4.6.1.2. Two LED Mode bits, one for each link, select whether the four LED signals corresponding to that link are controlled by software, or controlled by link state. If the LED signals are controlled by software, then the LED State bits control the value that is driven on those LED signals.

There are two modes of behavior for LED signals driven by link state. The mode is selected by sampling the PR_AD[3] signal either high or low at the deasserting edge of RST_L or LRST_L. If PR_AD[3] is sampled low, the LED signals reflect the receiver state of each differential pair. If PR_AD[3] is sampled high, the least significant LED signal in each group of four reflects the state of the link. The remaining LEDs are driven high. The LED signal state is shown in Table 3–40.

Table 3–40 Hardware-Controlled LED Signal State

LED Signal Pin State	LED Control S/W Mode	Differential Receiver Mode (PR_AD[3] sampled 0)	Link Mode (LEDx[0] only) (PR_AD[3] sampled 1)
High (LED off)	0	Differential receiver not synchronized or link not in Linked state	Link Down
Low (LED on)	1	Differential receiver synchronized, link in Linked state, and Traffic Enable set	Current link state is Linked and Traffic Enable set
≈ 500ms cycle (LED blinking)	–	Differential receiver synchronized, link in Linked state, and Traffic Enable not set	Current link state is Linked and Traffic Enable not set

4

Registers

The SG2010 implements the following registers:

- Bridge function configuration registers:

  PCI control, status, and address decoding registers for the Bridge
- Gateway function configuration registers:

  PCI control, status, and address decoding registers for the Gateway
- StarFabric Component (SFC) header registers:

  Fabric identity, control, and extended function list pointer registers
- CSRs:

  Control and status for fabric and device-specific functions

The SG2010 limits all register accesses from the PCI bus to single Dword transactions. The SG2010 permits multiple Dword write and prescriptive read request frames from the fabric, but restricts speculative reads to a single Dword.

The following general restrictions apply to register accesses:

- No register access during reset
- No register access during serial preload
- For leaf only, no address-routed register access from link when Lockout bit is set
- Frames must pass path protection checks, if enabled
- Channel 255 frames must pass range error checks

Note: *Register access is allowed during fabric enumeration.*

4.1 Register Address Spaces

Register access can occur through a PCI configuration read or write, a PCI memory read or write, a PCI I/O read or write, a Channel 255 read or write, or a serial ROM write. SG2010's registers are accessible using these mechanisms as shown in Table 4–1.

Table 4–1 SG2010 Register Access Mechanisms

Register Set	PCI Configuration Access	Channel 255 and PCI Memory (BAR0) Accesses	PCI I/O (BAR1) Access
Bridge Configuration Registers	Primary side when Bridge Enabled only	Dual-mapped at 5800h	Index to PCI memory offset
Gateway Configuration Registers	From PCI bus always; also Primary side if a leaf and Bridge Enabled	Dual-mapped at 5900h	Index to PCI memory offset
SFC Header and ELP Registers	No	Mapped starting at 0000h	Index to PCI memory offset
Functional CSRs	No	Mapped starting at 4000h	Some direct; others index to PCI memory offset

Dual-mapped means that the registers are directly accessible from that space, although possibly at a different register offset.

Indexed registers are used in I/O space to allow access to all registers. These Index registers are described in Section 4.1.3. In this case, a Channel 255 offset and data register are used in I/O space to get to the register locations. A limited set of Functional CSR registers are direct mapped into I/O space.

Serial ROM access is described in Section 4.3.

4.1.1 PCI Configuration Space Mappings

The SG2010 supports two PCI configuration register spaces, one for the Bridge function and one for the Gateway function. These register spaces are part of a multifunction device if the SG2010 is a root, and represent separate PCI devices if the SG2010 is a leaf. For more information about these configurations, see Section 3.1.1. The Bridge function is disabled when the Bridge Enable pin is pulled low. When the Bridge function is disabled, the Bridge configuration space is not accessible.

The configuration offsets are provided with their register descriptions in Sections 4.7 and 4.8. Neither Functional CSRs nor SFC Header registers are accessible in PCI configuration space.

4.1.2 PCI Memory Space Mappings

All the SG2010 registers are accessible through PCI memory space. PCI memory register offsets are the same as the Channel 255 offsets; however from PCI these are relative to the base address specified by BAR0. Table 4–2 is a summary of how this register space is mapped in PCI memory.

Table 4–2 PCI Memory Space Register Map Summary

Register Type	PCI Memory Offset	Mapping Notes
SFC Header and ELPs	0	
Functional CSRs	4000h	Functional CSRs except those noted below
Bridge Configuration	5800	Dual-mapped from Bridge Cfg offset 0
Gateway Configuration	5900	Dual-mapped from Gateway cfg offset 0
Functional CSRs	5C00	Source channel, Destination channel, Segment Table, and Path Table

The individual Functional CSR offsets are listed in Section 4.6.

4.1.3 PCI I/O Space Mappings

A subset of the SG2010 CSR registers are also directly accessible through PCI I/O space. However, all of the memory-mapped registers can be accessed through an indexing mechanism. Table 4–3 is a summary of how this register space is mapped.

Table 4–3 PCI I/O Space Register Map Summary

Register Type	PCI Memory Offset	Mapping Notes
CSRs	00h	Dual-mapped as shown in Table 4–8.
Indexes	30h	Offset/data index register pairs
CSRs	50h	Dual-mapped as shown in Table 4–8.

For a complete map of the I/O registers, see Table 4–11.

4.1.3.1 I/O Index Register Descriptions

The I/O index registers allow access to all of SG2010's registers from I/O space. Indexing is necessary because the maximum I/O space that can be requested is 256 bytes. The SG2010 implements four pairs of index registers. Each pair is composed of a Register Index register and a Register Data register. The Channel 255 offset of the desired target register is first written into an Register Index register, and the corresponding Register Data register is either read or written to read or write the target register, respectively.

4.1.3.1.1 I/O Index Offset x

Index Offset 0 I/O Byte Offset	0030h:0033h
Index Offset 1 I/O Byte Offset	0038h:003Bh
Index Offset 2 I/O Byte Offset	0040h:0043h
Index Offset 3 I/O Byte Offset	0048h:004Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
1:0	RES	R	0h	Reserved
13:2	INDOFFx	R/W	0h	I/O Index Offset. Contains the Channel 255 Dword-aligned offset of a register to be read or written using the indexing mechanism. Must be valid before an access to the I/O Index Data register is performed.
31:14	RES	R	0	Reserved

4.1.3.1.2 I/O Index Data x

Index Data 0 I/O Byte Offset	0034h:0037h
Index Data 1 I/O Byte Offset	003Ch:003Fh
Index Data 2 I/O Byte Offset	0044h:0047h
Index Data 3 I/O Byte Offset	004Ch:004FhSize4 byte

Bit	Name	Access	Reset Value	Description
31:0	INDDATAx	R/W	0h	I/O Index Data. When written using an I/O write operation, writes the data to the Channel 255 offset specified in the corresponding I/O Index Offset register. When read using an I/O read operation, reads the data from the CSR offset specified in the corresponding I/O Index Offset register. For both reads and writes, byte enables are used to enable byte access granularity.

4.1.4 Channel 255 Mappings

All the SG2010 registers are directly accessible in the Channel 255 address space. The standard StarFabric Component Header registers are located starting at offset 0. The offsets for these registers are listed in Section 4.5. Functional CSRs are accessible starting at offset 4000h.Table 4–4 lists the channel 255 mappings.

Table 4–4 Channel 255 Mappings

Register Type	Channel 255 Offset
SFC Header and ELP registers	0000h - 02FFh
Functional CSRs (see Table 4–8)	4000h - 57FFh
Bridge Configuration registers	5800h - 58FFh
Gateway Configuration registers	5900h - 59FFh
Functional CSRs (see Table 4–8)	5C00h - 7FFFh

If a Channel 255 access to a location at or above the 32KB boundary is received, then a range error occurs. A destination channel Range event is signaled and, if a response frame is returned, a Failure Type of Range Error is used.

4.2 Register Protection

The SG2010 has register access protection mechanisms to restrict register reads and writes. The SG2010 implements four path registers that can be enabled for register path protection. These path registers contain transformed paths (that is, the path from the SG2010 to the permissible origins of the frame), and an input port. If register path protection is enabled, any register access from the StarFabric interface, whether it uses Channel 255 path routing or address routing, is subject to path protection checks (Address-routed frames build a path as they travel through the fabric.). The path of the incoming frame is transformed and compared against all four path registers and their input ports. If any of the paths match, then the register access is performed. If none of the paths match, the register access is not performed. A Destination Channel Path Protection Error event bit is set, and if a response frame is required, a Channel Lock failure type is returned.

Each path protection register has an associated enable bit that enables the path protection compare against that registers. If none of the enable bits are set, then path protection is not performed for register accesses.

There are no protection mechanisms for local register access from the PCI bus.

Access of registers by a multicast write is not permitted. Multicast writes that specify Channel 255 result in a Channel Lock failure type (if a response frame is required) and a Destination Channel Path Protection event.

If the path protection check is successful, an address range comparison is performed. The Channel 255 address range is fixed to be the nearest 4KB boundary after the last implemented CSR and is not programmable. The SG2010 CSR range extends up to the 32KB boundary (7FFFh). The SG2010 does not perform the register access if the Channel 255 access has an offset addressing 8000h or higher. In this case, a Destination Channel Address Range Error event is signaled and if a response frame is required, a failure type of Range Error is returned.

4.3 SROM Preload

After chip reset is completed, the SG2010 performs a serial preload for register initialization. During serial preload, register access is not allowed. Accesses from the PCI bus are ignored (terminate in master abort). Accesses from the StarFabric interface result in a Lockout failure type if a response frame is required, and the incoming frame is dropped. No events are set.

Register preload data must start at byte offset 256 (100h) of the serial ROM. The SG2010 preloads the first byte of the preload data to detect the preload sequence enable, 10000b, in bits [7:3] of the first byte read. If the preload enable sequence is detected, then the fabric enumeration timer period is preloaded with the encoded value in bits [2:1] of that byte. If the preload sequence is not detected, the SG2010 terminates the SROM read and a preload is not performed.

The remainder of the preload allows register overloading and initialization. The register preload data consists of a list of preload operations. Each preload operation consists a Channel 255 offset, followed by a size field and a set of data bytes, where the number of data bytes is the same as the size field. The SG2010 preloads registers starting at the Channel 255 byte offset provided, and continuing until all the data bytes for that region are written. Subsequent preload operations in the list are then performed based on the Channel 255 offsets, sizes, and data byte provided. The SG2010 terminates the serial preload operation when a size field of 0 is detected.

Table 4–5 shows the preload data format for serial ROM.

Table 4–5 SROM Preload Data Format

Byte	Bits	Data
0	[7:3]	Preload sequence 10000b
	[2:1]	Fabric enumeration timer preload
	[0]	Reserved. Must be 0.
1	[7:0]	Channel 255 offset [7:0]
2	[6:0]	Channel 255 offset [14:8]
	[7]	Reserved. Must be 0.
3		Data field size *N* in bytes
Next *N* bytes		Data0 through Data*N-1*
		Repeat starting with Channel 255 offset [7:0] until data field size = 0 detected

Any register bit that has write access can be written through the SROM preload. Additionally, selected read-only registers, listed in Table 4–6, can also be overloaded through the preload.

Table 4–6 Read-only Registers with Preload Allowed

Register Space	Register Name	Bit name	Byte Offset	Bit Offset
GW Cfg	PCI Vendor ID	–	00h	–
	PCI Device ID	–	02h	–
	PCI Revision ID	–	08h	–
	Class Code	–	09h	–
	Subsystem Vendor ID	–	2Ch	–
	Subsystem Device ID	–	2Eh	–
	MIN_GNT	–	3Eh	–
	MAX_LAT	–	3Fh	–
Bridge Cfg	Slot Numbering Expansion Slot	–	4Eh	5:0
	Hot Swap Control	PI	5Ah	4
SFC Header	OEM Device Driver	–	8h	–
	StarFabric Protocol Revision	–	10h	–
	Programming Interface	–	1Ch	–

4.4 Register Maps

Table 4–7 maps the SFC header registers.

Table 4–7 StarFabric Component Header Register Map

Byte 3	Byte 2	Byte 1	Byte 0	CH255 Offset
SFC Vendor ID				0000h
SFC Device ID				0004h
SFC OEM Vendor Driver ID				0008h
Silicon Revision ID				000Ch
StarFabric Protocol Revision				0010h
SFC Base Class ID				0014h
Reserved				0018h
SFC Programming Interface				001Ch
Fabric ID				0020h
SFC Capabilities Register				0024h
Extended Function List Pointer (ELP)				0028h
SFC Control				002Ch
SFC Fabric Reset				0030h
Reserved				0034h – 003Fh
Semaphore ELP ID				0040h
Semaphore Next ELP				0044h
Semaphore Revision ID				0048h
Semaphore Offset Pointer				004Ch
Semaphore Number of Entries				0050h

Table 4–7 StarFabric Component Header Register Map *(Continued)*

Byte 3	Byte 2	Byte 1	Byte 0	CH255 Offset
Reserved				0054h – 005Fh
SGF ELP ID				0060h
SGF Next ELP				0064h
SGF Revision ID				0068h
SGF Offset Pointer				006Ch
Reserved				0070h – 007Fh
Port State Table ELP ID				0080h
Port State Table Next ELP				0084h
Port State Table Revision ID				0088h
Port State Table Offset Pointer				008Ch
Port State Table Number of Entries				0090h
Port State Table Entry Size				0094h
Reserved				0098h – 009Fh
Link State Table ELP ID				00A0h
Link State Table Next ELP				00A4h
Link State Table Revision ID				00A8h
Link State Table Offset Pointer				00ACh
Link State Table Number of Entries				00B0h
Link State Table Entry Size				00B4h
Reserved				00B8h – 00BFh
Event Table ELP ID				00C0h
Event Table Next ELP				00C4h
Event Table Revision ID				00C8h
Event Table Offset Pointer				00CCh
Reserved				00D0h – 00DFh
Port Map Table ELP ID				00E0h
Port Map Table Next ELP				00E4h
Port Map Table Revision ID				00E8h
Port Map Table Offset Pointer				00ECh
Port Map Table Number of Entries				00E0h
Port Map Table Entry Size				00E4h
Reserved				00E8h – 00EFh
Multicast ELP ID				0100h
Multicast Next ELP				0104h
Multicast Revision ID				0108h
Multicast Offset Pointer				010Ch
Multicast Number of Entries				0110h
Multicast Entry Size				0114h

Table 4–7 StarFabric Component Header Register Map *(Continued)*

Byte 3	Byte 2	Byte 1	Byte 0	CH255 Offset
Reserved				0118h – 010Fh
Segment Table ELP ID				0120h
Segment Table Next ELP				0124h
Segment Table Revision ID				0128h
Segment Table Offset Pointer				012Ch
Segment Table Number of Entries				0130h
Segment Table Entry Size				0134h
Reserved				0138h – 013Fh
Path Table ELP ID				0140h
Path Table Next ELP				0144h
Path Table Revision ID				0148h
Path Table Offset Pointer				014Ch
Path Table Number of Entries				0150h
Path Table Entry Size				0154h
Reserved				0158h – 015Fh
Destination Channel Table ELP ID				0160h
Destination Channel Table Next ELP				0164h
Destination Channel Table Revision ID				0168h
Destination Channel Table Offset Pointer				016Ch
Destination Channel Table Number of Entries				0170h
Destination Channel Table Entry Size				0174h
Reserved				0178h – 017Fh
Source Channel Table ELP ID				0180h
Source Channel Table Next ELP				0184h
Source Channel Table Revision ID				0188h
Source Channel Table Offset Pointer				018Ch
Source Channel Table Number of Entries				0190h
Source Channel Table Entry Size				0194h
Reserved				0198h – 019Fh
Channel 255 Path Protection ELP ID				01A0h
Channel 255 Path Protection Next ELP				01A4h
Channel 255 Path Protection Revision ID				01A8h
Channel 255 Path Protection Offset Pointer				01ACh
Channel 255 Path Protection Number of Entries				01B0h
Reserved				01B4h – 01BFh
Scratchpad ELP ID				01C0h
Scratchpad Next ELP				01C4h
Scratchpad Revision ID				01C8h

Table 4–7 StarFabric Component Header Register Map *(Continued)*

Byte 3	Byte 2	Byte 1	Byte 0	CH255 Offset
Scratchpad Offset Pointer				01CCh
Scratchpad Number of Entries				01D0h
Scratchpad Entry Size				01D4h
Reserved				01D8h – 01DFh
Channel 255 Registers ELP ID				01E0h
Channel 255 Registers Next ELP				01E4h
Channel 255 Registers Revision ID				01E8h
Channel 255 Registers Offset Pointer				01ECh
Channel 255 Registers Number of Entries				01F0h
Channel 255 Registers Entry Size				01F4h
Reserved				01F8h – 01FFh
VPD ELP ID				0200h
VPD Next ELP				0204h
VPD Revision ID				0208h
VPD Offset Pointer				020Ch
Reserved				0210h - 3FFFh

Table 4–8 maps the Functional Control and Status registers (CSRs).

Table 4–8 CSR Map

Byte 3	Byte 2	Byte 1	Byte 0	CH255 Offset
Chip Control and Status 0				4000h
Reserved				4004h
LED Control				4008h
Reserved				400Fh
Scratchpad 0 – Scratchpad 7				4010h – 402Fh
Reserved				4030h – 4053h
Doorbell Clear IRQ				4054h
Doorbell Clear IRQ Mask				4058h
Doorbell Set IRQ				405Ch
Doorbell Set IRQ Mask				4060h
Reserved				4064h – 40FFh
Port Map Table Entry 0 – Dword 0				4100h
Bridge Control		*Command*		
Port Map Table Entry 0 – Dword 1				4104h
Reserved	*Subordinate Bus #*	*Secondary Bus #*	*Primary Bus #*	
Port Map Table Entry 0 – Dword 2				4108h
Reserved		*I/O Limit*	*I/O Base*	

Table 4–8 CSR Map *(Continued)*

Byte 3	Byte 2	Byte 1	Byte 0	CH255 Offset
Port Map Table Entry 0 – Dword 3				410Ch
Memory Limit		*Memory base*		
Port Map Table Entry 0 – Dword 4				4110h
PF Memory Limit		*PF Memory Base*		
Port Map Table Entry 0 – Dword 5 – *PF Memory Base Upper 32 Bits*				4114h
Port Map Table Entry 0 – Dword 6 – *PF Memory Limit Upper 32 Bits*				4118h
Port Map Table Entry 0 – Dword 7				411Ch
I/O Limit Upper 16 Bits		*I/O Base Upper 16 Bits*		
Port Map Table Entry 1 – Dwords 0 through 7				4120h – 413Fh
Reserved				4140h – 41FFh
Multicast Group Table – Entries 0 through 31				4200h – 427Fh
Reserved				4280h – 43FFh
Semaphore 0 Clear				4400h
Semaphore 0 Set				4404h
Semaphore 0 Decrement				4408h
Semaphore 0 Increment				440Ch
Semaphore 0 Reserved 0				4410h
Semaphore 0 Increment if 0				4414h
Semaphore 0 Reserved 1				4418h
Semaphore 0 Increment if not 0				441Ch
Semaphore 1 (8 Dwords)				4420h – 443Fh
Semaphore 2 (8 Dwords)				4440h – 445Fh
Semaphore 3 (8 Dwords)				4460h – 447Fh
Semaphore 4 (8 Dwords)				4480h – 449Fh
Semaphore 5 (8 Dwords)				44A0h – 44BFh
Semaphore 6 (8 Dwords)				44C0h – 44DFh
Semaphore 7 (8 Dwords)				44E0h – 44FFh
Reserved				4500h – 4EFFh
Register Path Protection Path 0				4F00h
Register Path Protection Path 1				4F04h
Register Path Protection Path 2				4F08h
Register Path Protection Path 3				4F0Ch
Reserved				4F10h – 4F1Fh
Port 0 State Table – *Port to Link Map*				4F20h
Port 0 State Table – *Port Status*				4F24h
Port 1 State Table – *Port to Link Map*				4F28h
Port 1 State Table – *Port Status*				4F2Ch
Reserved				4F30h – 4F7Fh

Table 4–8 CSR Map *(Continued)*

Byte 3	Byte 2	Byte 1	Byte 0	CH255 Offset
Link 0 State Table – Dword 0 – *Control and Status*				4F80h
Link 0 State Table – Dword 1 – *Link Partner FID*				4F84h
Link 0 State Table – Dword 2				4F88h
8B/10B Error Count		*CRC Error Count*		
Link 0 State Table – Dword 3 – *Frame Count*				4F8Ch
Link 0 State Table – Dword 4 – *Line Count*				4F90h
Link 0 State Table – Dword 5 – *Empty Frame Count*				4F94h
Link 0 State Table – Dword 6				4F98h
Mcast Write Credit	*HP-Async Write Credit*	*Isoc Write Credit*	*Async Write Credit*	
Link 0 State Table – Dword 7				4F9Ch
Reserved	*Prov Write Credit*	*HP-Isoc Write Credit*	*Addr Write Credit*	
Link 0 State Table – Dword 8				4FA0h
T3 Write Credit	*T2 Write Credit*	*T1 Write Credit*	*T0 Write Credit*	
Link 0 State Table – Dword 9				4FA4h
T7 Write Credit	*T6 Write Credit*	*T5 Write Credit*	*T4 Write Credit*	
Link 0 State Table – Dword 10				4FA8h
Prov Req Credit	*HP-Isoc/Addr Req Credit*	*HP-Async Req Credit*	*Isoc/Async Req Credit*	
Link 0 State Table – Dword 11				4FACh
T7/6 Req Credit	*T5/4 Req Credit*	*T3/2 Req Credit*	*T1/0 Req Credit*	
Link 0 State Table – Dword 12				4FB0h
Reserved	*Diff. Pair State*	*Bandwidth Count*		
Link 0 State Table – Dword 12 – *Default CoS Credit Bytes 0-3*				4FB4h
Link 0 State Table – Dword 13 – *Default CoS Credit Bytes 4-7*				4FB8h
Link 0 State Table – Dword 14				4FBCh
Default Turn Request Credit	*Default Turn Write Credit*	*Default Cos Credit Bytes 8-9*		
Link 1 State Table – Dwords 0 through 14				4FC0h – 4FFFh
Chip Event Table – Dword 0				5000h
Chip Event Table Entry 1		*Chip Event Table Entry 0*		
Chip Event Table – Dwords 1 through 15 – *Entries 2 through 31*				5004h – 503Fh
Signal Event Table – Dword 0				5040h
Signal Event Table Entry 1		*Signal Event Table Entry 0*		
Signal Event Table – Dwords 1 through 6 – *Entries 2 through 13*				5044h – 505Bh
Reserved				505Ch – 507Fh
Event Path Table – Entry 0				5080h
Event Path Table – Entry 1				5084h

Table 4–8 CSR Map *(Continued)*

Byte 3	Byte 2	Byte 1	Byte 0	CH255 Offset
Event Path Table – Entry 2				5088h
Event Path Table – Entry 3				508Ch
Reserved				5090h – 509Fh
Event Mask Dword 0 Write-1-to-Clear				50A0h
Event Mask Dword 1 Write-1-to-Clear				50A4h
Reserved				50A8h – 50BFh
Event Mask Dword 0 Write-1-to-Set				50C0h
Event Mask Dword 1 Write-1-to-Set				50C4h
Reserved				50C8h – 50DFh
Raw Event Status Dword 0				50E0h
Raw Event Status Dword 1				50E4h
Reserved				50E8h – 50FFh
Event Status Dword 0				5100h
Event Status Dword 1				5104h
Reserved				5108h – 511Fh
Event Dispatch Control				5120h
Reserved				5124h – 51EFh
Event Handler Control				51F0h – 51F3h
Path Invalidation Control				51F4h – 51F7h
Event Buffer Upper Base Address				51F8h
Event Buffer Size				51FCh
EMU Address 0 – *EMU0 Counter Increment/Event Message Write*				5200h
EMU Address 1 – *EMU0 Counter Decrement*				5204h
EMU Address 2 – *EMU1 Counter Increment/Event Message Write*				5208h
EMU Address 3 – *EMU1 Counter Decrement*				520Ch
EMU Address 4 – *EMU2 Counter Increment/Event Message Write*				5210h
EMU Address 5 – *EMU2 Counter Decrement*				5214h
EMU Address 6 – *EMU3 Counter Increment/Event Message Write*				5218h
EMU Address 7 – *EMU3 Counter Decrement*				521Ch
EMU Address 8 – *EMU4 Counter Increment/Event Message Write*				5220h
EMU Address 9 – *EMU4 Counter Decrement*				5224h
EMU Address 10 – *EMU5 Counter Increment/Event Message Write*				5228h
EMU Address 11 – *EMU5 Counter Decrement*				522Ch
EMU Address 12 – *EMU6 Counter Increment/Event Message Write*				5230h
EMU Address 13 – *EMU6 Counter Decrement*				5234h
EMU Address 14 – *EMU7 Counter Increment/Event Message Write*				5238h
EMU Address 15 – *EMU7 Counter Decrement*				523Ch
EMU Address 16 – *EMU8 Counter Increment/Event Message Write*				5240h

Table 4–8 CSR Map *(Continued)*

Byte 3	Byte 2	Byte 1	Byte 0	CH255 Offset
EMU Address 17 – *EMU8 Counter Decrement*				5244h
Reserved				5248h – 527Fh
EMU0 – EMU8 Event Buffer Tail Pointers				5280h – 52A3h
Reserved				52A4h – 52BFh
EMU0 – EMU8 Event Buffer Lower Base Addresses				52C0h – 52E3h
Reserved				52E4h – 55FFh
SGF Frame Register – Dwords 0 through 35				5600h – 568Fh
Reserved				5690h – 5697h
SGF Destination Address – Dwords 0 and 1				5698h – 569Fh
SGF Control and Status				56A0h
SGF Bytes Received				56A4h
Reserved				56A8h - 56AFh
Manufacturing and Diagnostic Status (Undocumented)				56B0h
Reserved				56B4h – 57FFh
Bridge Configuration Registers (dual-map)				5800h – 58FFh
Gateway Configuration Registers (dual-map)				5900h – 59FFh
Reserved				5A00h – 5BFFh
Source Channel 0 – Dword 0 – *Source Channel Translation Address Lower*				5C00h
Source Channel 0 – Dword 1 – *Source Channel Translation Address Upper*				5C04h
Source Channel 0 – Dword 2 – *Source Channel Address Range Lower*				5C08h
Source Channel 0 – Dword 3				5C0Ch
Source Channel Control		*Source Channel Address Range Upper*		
Source Channel 1 through 7 – Dwords 0 through 3				5C10h – 5C7Fh
Reserved				5C80h – 5CFFh
Dest Channel 0 – Dword 0 – *Dest Channel Translation Address Lower*				5D00h
Dest Channel 0 – Dword 1 – *Dest Channel Translation Address Upper*				5D04h
Dest Channel 0 – Dword 2 – *Dest Channel Offset Range Lower*				5D08h
Dest Channel 0 – Dword 3				5D0Ch
Dest Channel Control		*Dest Channel Offset Range Upper*		
Dest Channel 0 – Dword 4 – *Channel Path Protection Path 0*				5D10h
Dest Channel 0 – Dword 5 – *Channel Path Protection Path 1*				5D14h
Dest Channel 0 – Dword 6 – *Reserved*				5D18h
Dest Channel 0 – Dword 7 – *Reserved*				5D1Ch
Dest Channels 1 through 7 – Dwords 0 through 7				5D20h – 5DFFh
Path Table – Entries 0 through 128				5E00h – 5FFFh
Segment Table Entry 0 – Dword 0				6000h
CoS/ Channel Control	*Last Turn/ Turn 0*	*Exit Port/ Path Length*	*Path Index/ Mcast ID*	

Table 4–8 CSR Map *(Continued)*

Byte 3	Byte 2	Byte 1	Byte 0	CH255 Offset
Segment Table Entry 0 – Dword 1				6004h
MSB Address		*Dest Channel ID*	*Src Channel ID*	
Segment Table – Entries 1 through 1023 – Dwords 0 and 1				6008h – 7FFFh

Table 4–9 maps the Bridge function PCI configuration registers.To derive the Channel 255 offset, add 5800h to the configuration offset.

Table 4–9 Bridge Configuration Register Map

Byte 3	Byte 2	Byte 1	Byte 0	Bridge Cfg Offset
Device ID		Vendor ID		00h
Status		Command		04h
Class Code			Revision ID	08h
Reserved	Header Type	PMLT	Cache Line Size	0Ch
Reserved				10h
Reserved				14h
SMLT	Subordinate Bus #	Secondary Bus #	Primary Bus #	18h
Secondary Status		I/O Limit	I/O Base	1Ch
Memory Limit		Memory Base		20h
PF Memory Limit		PF Memory Base		24h
PF Memory Base Upper 32 Bits				28h
PF Memory Limit Upper 32 Bits				2Ch
I/O Limit Upper 16 Bits		I/O Base Upper 16 Bits		30h
Reserved			ECP	34h
Reserved				38h
Bridge Control		Interrupt Pin	Interrupt Line	3Ch
Reserved				40h
PM Capabilities		PM Next ECP	PM ECP ID	44h
PM Data	PM P2P Support	PM Control and Status		48h
Slot # Chassis #	Slot # Exp Slot	Slot # Next ECP	Slot # ECP ID	4Ch
VPD Address		VPD Next ECP	VPD ECP ID	50h
VPD Data				54h
Hot Swap Control		HS Next ECP	HS ECP ID	58h
Reserved				5Ch - FFh

Table 4–10 maps the Gateway function PCI configuration registers. To derive the Channel 255 offset, add 5900h to the configuration offset.

Table 4–10 Gateway Configuration Register Map

Byte 3	Byte 2	Byte 1	Byte 0	GW Cfg Offset
Device ID		Vendor ID		00h
Status		Command		04h
Class Code			Revision ID	08h
Reserved	Header Type	MLT	Cache Line Size	0Ch
BAR0				10h
BAR1				14h
BAR2				18h
BAR3				1Ch
BAR4				20h
BAR5				24h
Reserved				28h
Subsystem ID		Subsystem Vendor ID		2Ch
Expansion ROM BAR				30h
Reserved			ECP	34h
Reserved				38h
MAX_LAT	MIN_GNT	Interrupt Pin	Interrupt Line	3Ch
SGT Configuration Address				40h
SGT Configuration Data				44h
SGT I/O Address				48h
SGT I/O Data				4Ch
Reserved				50h – 57h
ROM Setup				58h
ROM Address				5Ch
ROM Data				60h
Secondary Limit		Secondary Base		64h
Secondary Base Upper 32 Bits				68h
Secondary Limit Upper 32 Bits				6Ch
IDSEL Mask				70h
Reserved				74h – 7Fh
BAR2 Setup				80h
BAR3 Setup				84h
BAR4 Setup				88h
BAR5 Setup				8Ch
Expansion ROM BAR Setup				90h
Chip Control				94h
Arbiter Control		Chip Status		98h

Table 4–10 Gateway Configuration Register Map *(Continued)*

Byte 3	Byte 2	Byte 1	Byte 0	GW Cfg Offset
Reserved				9Ch
GPIO Dir Clear	GPIO Data Clear	GPIO Dir Set	GPIO Data Set	A0h
Reserved				A4h – ABh
PM Capabilities		PM Next ECP	PM ECP ID	ACh
PM Data	PM P2P Support	PM Control and Status		B0h
VPD Address		VPD Next ECP	VPD ECP ID	B4h
VPD Data				B8h
MSI Message Control		MSI Next ECP	MSI ECP ID	BCh
MSI Message Address				C0h
MSI Message Address Upper 32 Bits				C4h
Reserved		MSI Message Data		C8h
Reserved				CCh - D7h
Hot Swap Control (Gateway only mode)		HS Next ECP (GW only mode)	HS ECP ID (GW only mode)	D8h
Reserved				DCh - FFh

Table 4–11 maps the I/O registers.

Table 4–11 I/O Register Map

Byte 3	Byte 2	Byte 1	Byte 0	I/O Byte Offset
Chip Control and Status 0				00h
Reserved				04h
Reserved				08h – 0Fh
Scratchpad 0 – Scratchpad 7				10h – 2Fh
I/O Index Offset 0				30h
I/O Index Data 0				34h
I/O Index Offset 1				38h
I/O Index Data 1				3Ch
I/O Index Offset 2				40h
I/O Index Data 2				44h
I/O Index Offset 3				48h
I/O Index Data 3				4Ch
Reserved				50h
Doorbell Clear IRQ				54h
Doorbell Clear IRQ Mask				58h
Doorbell Set IRQ				5Ch
Doorbell Set IRQ Mask				60h
Reserved				64h – 7Fh

4.5 StarFabric Component (SFC) Header Registers

Note: *Register Access field abbreviations are defined in the Conventions section of the Preface.*

4.5.1 SFC Header

These registers are defined for every StarFabric device.

4.5.1.1 SFC Vendor ID

Ch. 255 Byte Offset	0000h:0003h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	FVENDORID	R	1h	Returns 01h, the StarFabric Vendor ID for StarGen.

4.5.1.2 SFC Device ID

Ch. 255 Byte Offset	0004h:0007h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	FDEVICEID	R	1h	Returns 01h, the StarFabric Device ID for the SG2010.

4.5.1.3 OEM Vendor Driver ID

Ch. 255 Byte Offset	0008h:000Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	FOEMDRID	R	0h	Returns the OEM Driver ID for the SG2010, which is initialized to 0, but loadable through serial ROM preload.

4.5.1.4 Silicon Revision

Ch. 255 Byte Offset	000Ch:000Fh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	SILREVID	R	H/W*	Returns the silicon revision ID for the SG2010.

* H/W = Determined by hardware. Changes with every silicon revision.

4.5.1.5 StarFabric Protocol Revision

Ch. 255 Byte Offset	0010h:0013h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	FPREVID	R	1h	Returns the Protocol Revision ID for the SG2010, which is initialized to 1, but loadable through serial ROM preload.

4.5.1.6 SFC Base Class ID

Ch. 255 Byte Offset	0014h:0017h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
7:0	SBASECL	R	0h	Identifies the switch class. Reads as 0 to indicate that this device does not perform switching.
15:8	EBASECL	R	1h	Identifies the edge class. Reads as 1 to indicate that this is a bridge to another protocol.
31:16	ESUBCL	R	1h	Identifies the edge subclass. Reads as 1 to indicate that this device bridges between StarFabric protocol and the PCI protocol.

4.5.1.7 SFC Programming Interface ID

Ch. 255 Byte Offset	001Ch:001Fh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	FPROGIF	R	0h	Identifies the programming interface for the device. Initialized to 0, but loadable through serial ROM.

4.5.1.8 Fabric ID (FID)

Fabric ID for the device. This value is determined during fabric enumeration and is also the path from the root node.

Ch. 255 Byte Offset	0020h:0023h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
4:0	RES	R	0h	Reserved
7:5	PFN	R	111h	Fabric ID Parallel Fabric Number.
10:8	TC	R	111h	Fabric ID Turn Count. Identifies the number of turns from the root note to this device.
13:11	TURN0	R	111h	Fabric ID Turn 0. Identifies Turn 0 on the path from the root node to this node.

16:14	TURN1	R	111h	Fabric ID Turn 1. Identifies Turn 1 on the path from the root node to this node.
19:17	TURN2	R	111h	Fabric ID Turn 2. Identifies Turn 2 on the path from the root node to this node.
22:20	TURN3	R	111h	Fabric ID Turn 3. Identifies Turn 3 on the path from the root node to this node.
25:23	TURN4	R	111h	Fabric ID Turn 4. Identifies Turn 4 on the path from the root node to this node.
28:26	TURN5	R	111h	Fabric ID Turn 5. Identifies Turn 5 on the path from the root node to this node.
31:29	TURN6	R	111h	Fabric ID Turn 6. Identifies Turn 6 on the path from the root node to this node.

4.5.1.9 SFC Capabilities

Ch. 255 Byte Offset	0024h:0027h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
0	ARCAP	R	H/W	Indicates whether the SG2010 has address-routing capabilities. If the Bridge function is enabled, this bit reads as 1 indicating address routing is supported. If the Bridge function is disabled, this bit reads as 0 indicating address routing is not supported.
7:1	RES	R	0	Reserved
15:8	CRSHR	R	0	Credit sharing capabilities. Reads as 0 to indicate that credits may be reallocated between different CoS and between CoS and turn credits.
31:1	RES	R	0	Reserved.

4.5.1.10 Extended Function List Pointer (ELP)

Ch. 255 Byte Offset	0028h:002Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPPTR	R	40h	Channel 255 offset pointer to the first ELP list element.

4.5.1.11 SFC Control

Ch. 255 Byte Offset	002Ch:002Fh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
0	NRESET	R/W	0h	Node reset. When written with 1, the chip is reset. This reset does not propagate to the fabric, but the SG2010 does assert RSTO_L for a minimum of 100μsec. The SG2010 clears this bit to 0 when reset is complete. If the Propagate Maskable Reset bit is also written with a 1 during the same write access, a maskable reset comma is propagated out all links prior to the chip reset.
1	FRSTENA	R/W1TC	1h	Fabric reset enable. When 1, enables a fabric reset when the Fabric Reset bit is set. When 0, a write to the Fabric Reset bit has no effect. After this bit is cleared by a write-1-to-clear operation, only a chip reset can set it back to 1.
2	PRPRST	WRZ	0h	Propagate maskable reset. When written with a 1, a maskable reset comma is propagated out all the links. The chip is subsequently reset if the Node Reset bit is also written with a 1 in the same access. If the Node Reset bit is written with a 0 when this bit is written with a 1, then no chip reset is performed after the reset propagation. This bit always returns 0 when read.
31:2	RES	R	0	Reserved

4.5.1.12 Fabric Reset

Ch. 255 Byte Offset	0030h:0033h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
0	FRESET	R/W	0h	Fabric reset. When written with 1, and if the Fabric Reset Enable bit is 1, the SG2010 propagates an unmaskable reset into the fabric and then performs a chip reset. The SG2010 clears this bit to 0 when chip reset is complete.
31:1	RES	R	0	Reserved

4.5.2 Extended Function List Elements

These registers comprise a linked list of pointers to registers controlling StarFabric functions. A summary of the values used by the SG2010's ELPs is provided in Table 4–12.

Table 4–12 Extended List Pointer (ELP) Summary

Function	ELP Offset	ELP ID	Offset Pointer	Number of Entries	Entry Size	Next ELP Offset
Semaphores	0040h	08h	4400h	08h	N/A	0060h
SGF	0060h	(1)07h	5600h	N/A	N/A	0080h
Port State Table	0080h	0Ch	4F20h	02h	08h	00A0h
Link State Table	00A0h	05h	4F80h	02h	40h	00C0h
Events	00C0h	06h	5000h	N/A	N/A	00E0h
Port Map Table	00E0h	0Dh	4100h	02h	20h	0100h
Multicast Table	0100h	0Ah	4200h	20h	04h	0120h
Segment Table	0120h	(1)01h	6000h	0400h	08h	0140h
Path Table	0140h	(1)02h	5E00h	80h	04h	0160h
Destination Channels	0160h	03h	5D00h	08h	20h	0180h
Source Channels	0180h	(1)04h	5C00h	08h	10h	01A0h
Ch255 Path Protection	01A0h	09h	4F00h	04h	N/A	01C0h
Scratchpad	01C0h	0Eh	4010h	08h	04h	01E0h
Channel 255 Registers	01E0h	00h	0000h	01h	8000h	0200h
VPD	0200h	0Fh	59B4h	N/A	N/A	0000h

4.5.2.1 Semaphore ELP

4.5.2.1.1 Semaphore ELP ID

Ch. 255 Byte Offset	0040h:0043h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
30:0	ELPID8	R	8h	Returns the Semaphore ELP ID.
31	ELPIDM8	R	0h	Reads as a 0 to indicate that this ELP is defined by the StarFabric protocol.

4.5.2.1.2 Semaphore ELP Next ELP

Ch. 255 Byte Offset	0044h:0047h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNXT8	R	60h	Returns the pointer to the next ELP Channel 255 offset.

4.5.2.1.3 Semaphore ELP Revision ID

Ch. 255 Byte Offset	0048h:004Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
30:0	ELPREV8	R	0h	Returns the revision number of this Semaphore implementation.

4.5.2.1.4 Semaphore ELP Offset Pointer

Ch. 255 Byte Offset	004Ch:004Fh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPOFF8	R	4400h	Returns the Channel 255 offset of the Semaphore registers.

4.5.2.1.5 Semaphore ELP Number

Ch. 255 Byte Offset	0050h:0053h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNUM8	R	8h	Returns the number of semaphores implemented by the SG2010.

4.5.2.2 SGF ELP

4.5.2.2.1 SGF ELP ID

Ch. 255 Byte Offset	0060h:0063h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
30:0	ELPID7	R	7h	Returns the SGF ELP ID.
31	ELPIDM7	R	1h	Reads as 1 to indicate that this ELP points to a device-specific function.

4.5.2.2.2 SGF ELP Next ELP

Ch. 255 Byte Offset	0064h:0067h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNXT7	R	80h	Returns the pointer to the next ELP Channel 255 offset.

4.5.2.2.3 SGF ELP Revision ID

Ch. 255 Byte Offset	0068h:006Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPREV7	R	0h	Returns the revision number of this SGF implementation.

4.5.2.2.4 SGF ELP Offset Pointer

Ch. 255 Byte Offset	006Ch:006Fh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPOFF7	R	5600h	Returns the Channel 255 offset of the SGF registers.

4.5.2.3 Port State Table ELP

4.5.2.3.1 Port State ELP ID

Ch. 255 Byte Offset	0080h:0083h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
30:0	ELPIDC	R	Ch	Returns the Port State ELP ID.
31	ELPIDMC	R	0h	Reads as a 0 to indicate that this ELP is defined by the StarFabric protocol.

4.5.2.3.2 Port State ELP Next ELP

Ch. 255 Byte Offset	0084h:0087h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNXTC	R	A0h	Returns the pointer to the next ELP Channel 255 offset.

4.5.2.3.3 Port State ELP Revision ID

Ch. 255 Byte Offset	0088h:008Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPREVC	R	0h	Returns the revision number of this Port State implementation.

4.5.2.3.4 Port State ELP Offset Pointer

Ch. 255 Byte Offset	008Ch:008Fh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPOFFC	R	4F20h	Returns the Channel 255 offset of the Port State registers.

4.5.2.3.5 Port State ELP Number

Ch. 255 Byte Offset	0090h:0093h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNUMC	R	2h	Returns the number of port state entries implemented by the SG2010.

4.5.2.3.6 Port State ELP Entry Size

Ch. 255 Byte Offset	0094h:0097h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPSIZC	R	8h	Returns the size in bytes of each port state entry.

4.5.2.4 Link State Table ELP

4.5.2.4.1 Link State ELP ID

Ch. 255 Byte Offset	00A0h:00A3h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
30:0	ELPID5	R	5h	Returns the Link State ELP ID.
31	ELPIDM5	R	0h	Reads as 0 to indicate that this ELP is defined by the StarFabric protocol.

4.5.2.4.2 Link State ELP Next ELP

Ch. 255 Byte Offset	00A4h:00A7h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNXT5	R	C0h	Returns the pointer to the next ELP Channel 255 offset.

4.5.2.4.3 Link State ELP Revision ID

Ch. 255 Byte Offset	00A8h:00ABh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPREV5	R	0h	Returns the revision number of this link state implementation.

4.5.2.4.4 Link State ELP Offset Pointer

Ch. 255 Byte Offset	00ACh:00AFh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPOFF5	R	4F80h	Returns the Channel 255 offset of the Link State registers.

4.5.2.4.5 Link State ELP Number

Ch. 255 Byte Offset	00B0h:00B3h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNUM5	R	2h	Returns the number of link state entries implemented by the SG2010.

4.5.2.4.6 Link State ELP Entry Size

Ch. 255 Byte Offset	00B4h:00B7h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPSIZ5	R	40h	Returns the size in bytes of each link state entry.

4.5.2.5 Event ELP

4.5.2.5.1 Event ELP ID

Ch. 255 Byte Offset	00C0h:00C3h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
30:0	ELPID6	R	6h	Returns the Event ELP ID.
31	ELPIDM6	R	0h	Reads as 0 to indicate that this ELP is defined by the StarFabric protocol.

4.5.2.5.2 Event ELP Next ELP

Ch. 255 Byte Offset	00C4h:00C7h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNXT6	R	E0h	Returns the pointer to the next ELP Channel 255 offset.

4.5.2.5.3 Event ELP Revision ID

Ch. 255 Byte Offset	00C8h:00CBh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPREV6	R	0h	Returns the revision number of this event implementation.

4.5.2.5.4 Event ELP Offset Pointer

Ch. 255 Byte Offset	00CCh:00CFh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPOFF6	R	5000h	Returns the Channel 255 offset of the Event registers.

4.5.2.6 Port Map ELP

4.5.2.6.1 Port Map ELP ID

Ch. 255 Byte Offset	00E0h:00E3h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
30:0	ELPIDD	R	Dh	Returns the Port Map ELP ID.
31	ELPIDMD	R	0h	Reads as 0 to indicate that this ELP is defined by the StarFabric protocol.

4.5.2.6.2 Port Map ELP Next ELP

Ch. 255 Byte Offset	00E4h:00E7h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNXTD	R	100h	Returns the pointer to the next ELP Channel 255 offset.

4.5.2.6.3 Port Map ELP Revision ID

Ch. 255 Byte Offset	00E8h:00EBh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPREVD	R	0h	Returns the revision number of this Port Map implementation.

4.5.2.6.4 Port Map ELP Offset Pointer

Ch. 255 Byte Offset	00ECh:00EFh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPOFFD	R	4100h	Returns the Channel 255 offset of the Port Map registers.

4.5.2.6.5 Port Map ELP Number

Ch. 255 Byte Offset	00F0h:00F3h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNUMD	R	2h	Returns the number of Port Map entries implemented by the SG2010.

4.5.2.6.6 Port Map ELP Entry Size

Ch. 255 Byte Offset	00F4h:00F7h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPSIZD	R	20h	Returns the size in bytes of each Port Map entry.

4.5.2.7 Multicast ELP

4.5.2.7.1 Multicast ELP ID

Ch. 255 Byte Offset	0100h:0103h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
30:0	ELPIDA	R	Ah	Returns the Multicast ELP ID.
31	ELPIDMA	R	0h	Reads as 0 to indicate that this ELP is defined by the StarFabric protocol.

4.5.2.7.2 Multicast ELP Next ELP

Ch. 255 Byte Offset	0104h:0107h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNXTA	R	120h	Returns the pointer to the next ELP Channel 255 offset.

4.5.2.7.3 Multicast ELP Revision ID

Ch. 255 Byte Offset	0108h:010Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPREVA	R	0h	Returns the revision number of this Multicast implementation.

4.5.2.7.4 Multicast ELP Offset Pointer

Ch. 255 Byte Offset	010Ch:010Fh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPOFFA	R	4200h	Returns the Channel 255 offset of the Multicast registers.

4.5.2.7.5 Multicast ELP Number

Ch. 255 Byte Offset	0110h:0113h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNUMA	R	20h	Returns the number of Multicast entries implemented by the SG2010.

4.5.2.7.6 Multicast ELP Entry Size

Ch. 255 Byte Offset	0114h:0117h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPSIZA	R	4h	Returns the size in bytes of each Multicast entry.

4.5.2.8 Segment Table ELP

4.5.2.8.1 Segment Table ELP ID

Ch. 255 Byte Offset	0120h:0123h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
30:0	ELPID1	R	1h	Returns the Segment Table device-specific ELP ID.
31	ELPIDM1	R	1h	Reads as 1 to indicate this ELP is device-specific.

4.5.2.8.2 Segment Table ELP Next ELP

Ch. 255 Byte Offset	0124h:0127h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNXT1	R	140h	Returns the pointer to the next ELP Channel 255 offset.

4.5.2.8.3 Segment Table ELP Revision ID

Ch. 255 Byte Offset	0128h:012Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPREV1	R	0h	Returns the revision number of this Segment Table implementation.

4.5.2.8.4 Segment Table ELP Offset Pointer

Ch. 255 Byte Offset	012Ch:012Fh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPOFF1	R	6000h	Returns the Channel 255 offset of the Segment Table registers.

4.5.2.8.5 Segment Table ELP Number

Ch. 255 Byte Offset	0130h:0133h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNUM1	R	400h	Returns the number of Segment Table entries implemented by the SG2010.

4.5.2.8.6 Segment Table ELP Entry Size

Ch. 255 Byte Offset	0134h:0137h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPSIZ1	R	8h	Returns the size in bytes of each Segment Table entry.

4.5.2.9 Path Table ELP

4.5.2.9.1 Path Table ELP ID

Ch. 255 Byte Offset	0140h:0143h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
30:0	ELPIDA	R	2h	Returns the Path Table device-specific ELP ID.
31	ELPIDMA	R	1h	Reads as 1 to indicate this ELP is device-specific.

4.5.2.9.2 Path Table ELP Next ELP

Ch. 255 Byte Offset	0144h:0147h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNXTA	R	160h	Returns the pointer to the next ELP Channel 255 offset.

4.5.2.9.3 Path Table ELP Revision ID

Ch. 255 Byte Offset	0148h:014Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPREV2	R	0h	Returns the revision number of this Path Table implementation.

4.5.2.9.4 Path Table ELP Offset Pointer

Ch. 255 Byte Offset	014Ch:014Fh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPOFF2	R	5E00h	Returns the Channel 255 offset of the Path Table registers.

4.5.2.9.5 Path Table ELP Number

Ch. 255 Byte Offset	0150h:0153h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNUM2	R	80h	Returns the number of Path Table entries implemented by the SG2010.

4.5.2.9.6 Path Table ELP Entry Size

Ch. 255 Byte Offset	0154h:0157h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPSIZ2	R	4h	Returns the size in bytes of each Path Table entry.

4.5.2.10 Destination Channel ELP

4.5.2.10.1 Destination Channel ELP ID

Ch. 255 Byte Offset	0160h:0163h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
30:0	ELPID3	R	3h	Returns the Destination Channel ELP ID.
31	ELPIDM3	R	0h	Reads as 0 to indicate that this ELP is defined by the StarFabric protocol.

4.5.2.10.2 Destination Channel ELP Next ELP

Ch. 255 Byte Offset	0164h:0167h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNXT3	R	0180h	Returns the pointer to the next ELP Channel 255 offset.

4.5.2.10.3 Destination Channel ELP Revision ID

Ch. 255 Byte Offset	0168h:016Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPREV3	R	0h	Returns the revision number of this Destination Channel implementation.

4.5.2.10.4 Destination Channel ELP Offset Pointer

Ch. 255 Byte Offset	016Ch:016Fh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPOFF3	R	5D00h	Returns the Channel 255 offset of the Destination Channel registers.

4.5.2.10.5 Destination Channel ELP Number

Ch. 255 Byte Offset	0170h:0173h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNUM3	R	8h	Returns the number of Destination Channel entries implemented by the SG2010.

4.5.2.10.6 Destination Channel ELP Entry Size

Ch. 255 Byte Offset	0174h:0177h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPSIZ3	R	20h	Returns the size in bytes of each Destination Channel entry.

4.5.2.11 Source Channel ELP

4.5.2.11.1 Source Channel ELP ID

Ch. 255 Byte Offset	0180h:0183h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
30:0	ELPID4	R	4h	Returns the Source Channel ELP ID.
31	ELPIDM4	R	1h	Reads as 1 to indicate that this ELP is device-specific

4.5.2.11.2 Source Channel ELP Next ELP

Ch. 255 Byte Offset	0184h:0187h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNXT4	R	01A0h	Returns the pointer to the next ELP Channel 255 offset.

4.5.2.11.3 Source Channel ELP Revision ID

Ch. 255 Byte Offset	0188h:018Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPREV4	R	0h	Returns the revision number of this Source Channel implementation.

4.5.2.11.4 Source Channel ELP Offset Pointer

Ch. 255 Byte Offset	018Ch:018Fh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPOFF4	R	5C00h	Returns the Channel 255 offset of the Source Channel registers.

4.5.2.11.5 Source Channel ELP Number

Ch. 255 Byte Offset	0190h:0193h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNUM4	R	8h	Returns the number of Source Channel entries implemented by the SG2010.

4.5.2.11.6 Source Channel ELP Entry Size

Ch. 255 Byte Offset	0194h:0197h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPSIZ4	R	10h	Returns the size in bytes of each Source Channel entry.

4.5.2.12 Channel 255 Path Protection ELP

4.5.2.12.1 Channel 255 Path Protection ELP ID

Ch. 255 Byte Offset	01A0h:01A3h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
30:0	ELPID9	R	9h	Returns the Channel 255 Path Protection ELP ID.
31	ELPIDM9	R	0h	Reads as 0 to indicate that this ELP is defined by the StarFabric protocol.

4.5.2.12.2 Channel 255 Path Protection ELP Next ELP

Ch. 255 Byte Offset	01A4h:01A7h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNXT9	R	01C0h	Returns the pointer to the next ELP Channel 255 offset.

4.5.2.12.3 Channel 255 Path Protection ELP Revision ID

Ch. 255 Byte Offset	01A8h:01ABh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPREV9	R	0h	Returns the revision number of this Channel 255 Path Protection implementation.

4.5.2.12.4 Channel 255 Path Protection ELP Offset Pointer

Ch. 255 Byte Offset	01ACh:01AFh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPOFF9	R	4F00h	Returns the Channel 255 offset of the Channel 255 Path Protection registers.

4.5.2.12.5 Channel 255 Path Protection ELP Number

Ch. 255 Byte Offset	01B0h:01B3h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNUM9	R	4h	Returns the number of Channel 255 Path Protection entries implemented by the SG2010.

4.5.2.13 Scratchpad ELP

4.5.2.13.1 Scratchpad ELP ID

Ch. 255 Byte Offset	01C0h:01C3h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
30:0	ELPIDE	R	Eh	Returns the Scratchpad ELP ID.
31	ELPIDME	R	0h	Reads as 0 to indicate that this ELP is defined by the StarFabric protocol.

4.5.2.13.2 Scratchpad Next ELP

Ch. 255 Byte Offset	01C4h:01C7h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNXTE	R	01E0h	Returns the pointer to the next ELP Channel 255 offset.

4.5.2.13.3 Scratchpad ELP Revision ID

Ch. 255 Byte Offset	01C8h:01CBh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPREVE	R	0h	Returns the revision number of this Channel 255 Path Protection implementation.

4.5.2.13.4 Scratchpad Offset Pointer

Ch. 255 Byte Offset	01CCh:01CFh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPOFFE	R	4010h	Returns the Channel 255 offset of the Scratchpad registers.

4.5.2.13.5 Scratchpad ELP Number

Ch. 255 Byte Offset	01D0h:01D3h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNUME	R	8h	Returns the number of scratchpad registers implemented by the SG2010.

4.5.2.13.6 Scratchpad ELP Entry Size

Ch. 255 Byte Offset	01D4h:01D7h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPSIZE	R	4h	Returns the size in bytes of each scratchpad entry.

4.5.2.14 Channel 255 Registers ELP

4.5.2.14.1 Channel 255 Registers ELP ID

Ch. 255 Byte Offset	01E0h:01E3h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
30:0	ELPID0	R	0h	Returns the Channel 255 Registers ELP ID.
31	ELPIDM0	R	0h	Reads as 0 to indicate that this ELP is defined by the Star-Fabric protocol.

4.5.2.14.2 Channel 255 Registers Next ELP

Ch. 255 Byte Offset	01E4h:01E7h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNXT0	R	0200h	Returns the pointer to the next ELP Channel 255 offset.

4.5.2.14.3 Channel 255 Registers ELP Revision ID

Ch. 255 Byte Offset	01E8h:01EBh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPREV0	R	0h	Returns the revision number of this Channel 255 Register map.

4.5.2.14.4 Channel 255 Registers Offset Pointer

Ch. 255 Byte Offset	01ECh:01EFh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPOFF0	R	0h	Returns the Channel 255 offset of the Channel 255 registers.

4.5.2.14.5 Channel 255 Registers ELP Number

Ch. 255 Byte Offset	01F0h:01F3h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNUM0	R	1h	Reads as 1.

4.5.2.14.6 Channel 255 Registers ELP Entry Size

Ch. 255 Byte Offset	01F4h:01F7h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPSIZ0	R	8000h	Returns the size in bytes of the Channel 255 register map.

4.5.2.15 Vital Product Data (VPD) ELP

4.5.2.15.1 VPD ELP ID

Ch. 255 Byte Offset	0200h:0203h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
30:0	ELPIDF	R	Fh	Returns the VPD ELP ID.
31	ELPIDMF	R	0h	Reads as 0 to indicate that this ELP is defined by the Star-Fabric protocol.

4.5.2.15.2 VPD Next ELP

Ch. 255 Byte Offset	01E4h:01E7h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPNXTF	R	0h	Returns the pointer to the next ELP Channel 255 offset This is the last pointer.

4.5.2.15.3 VPD ELP Revision ID

Ch. 255 Byte Offset	01E8h:01EBh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPREVF	R	0h	Returns the revision number of this VPD registers.

4.5.2.15.4 VPD Offset Pointer

Ch. 255 Byte Offset	01ECh:01EFh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELPOFFF	R	59B4h	Returns the Channel 255 offset of the VPD registers.

4.6 Control and Status Registers (CSRs)

The CSRs are mapped in Channel 255 space starting at the 16K byte boundary (4000h), and in PCI memory space starting at byte 0 of the BAR0 base address.

4.6.1 Device Specific Functions and Control

4.6.1.1 Chip Control Status 0

Ch. 255 Byte Offset	4000h:4003h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
16:0	RES	R	0	Reserved
17	BUNDLED	R	0	When 0, indicates that SG2010’s two links are separate ports. When 1, indicates that SG2010’s two links are bundled and comprise a single port.
19:18	RES	R	0	Reserved.
20	BAR2RED	R/W	0	BAR2 Redundant Route Enable. When 0, indicates that the SG2010 does not have redundant paths for the BAR2 address range. When 1, indicates that the SG2010 has been configured to support redundant paths for the BAR2 address range. This bit is meaningless when BAR2 and BAR3 are combined to enable a 64-bit BAR.
21	BAR3RED	R/W	0	BAR3 Redundant Route Enable. When 0, indicates that the SG2010 does not have redundant paths for the BAR3 address range. When 1, indicates that the SG2010 has been configured to support redundant paths for the BAR3 address range. This bit is also used when BAR2 and BAR3 are combined to enable a 64-bit BAR.
22	BAR4RED	R/W	0	BAR4 Redundant Route Enable. When 0, indicates that the SG2010 does not have redundant paths for the BAR4 address range. When 1, indicates that the SG2010 has been configured to support redundant paths for the BAR4 address range. This bit is meaningless when BAR4 and BAR5 are combined to enable a 64-bit BAR.
23	BAR5RED	R/W	0	BAR5 Redundant Route Enable. When 0, indicates that the SG2010 does not have redundant paths for the BAR5 address range. When 1, indicates that the SG2010 has been configured to support redundant paths for the BAR5 address range. This bit is also used when BAR4 and BAR5 are combined to enable a 64-bit BAR.
31:24	RES	R	0	Reserved

4.6.1.2 LED Control

Ch. 255 Byte Offset	4008h:400Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
3:0	LED0S	R/W	0	LED0[3:0] S/W State. When the LED0_L Mode bit is 1 and a bit is: 0 The corresponding LED0_L[3:0] signal is driven high, turning off the LED. 1 The corresponding LED0_L[3:0] signal is driven low, turning on the LED.
7:4	LED1S	R/W	0	LED1[3:0] S/W State. When the LED1_L Mode bit is 1 and a bit is: 0 The corresponding LED1_L[3:0] signal is driven high, turning off the LED. 1 The corresponding LED1_L[3:0] signal is driven low, turning on the LED.
23:8	RES	R	0	Reserved
24	LED0C	R/W	0	LED0_L Control Mode. Selects either H/W or S/W control for driving LED0. When: 0 The SG2010 hardware drives the LED0_L[3:0] signal pins with either link or differential pair receiver state. 1 The SG2010 drives the LED0_L[3:0] signal pins as specified by the LED0S software state.
25	LED1C	R/W	0	LED1_L Control Mode. Selects either H/W or S/W control for driving LED1. When: 0 The SG2010 hardware drives the LED1_L[3:0] signal pins with either link or differential pair receiver state. 1 The SG2010 drives the LED1_L[3:0] signal pins as specified by the LED1S software state.
29:26	RES	R	0	Reserved.

Bit	Name	Access	Reset Value	Description
30	LEDHSS	R/W1TS	H/W	LED Hardware State Set. Selects either link or differential pair state to be reflected by the LEDs. When read, this bit reflects the value of the LED H/W State. When the LED Hardware State is: 0 Differential pair receiver state is driven on all eight LED signals if the corresponding LEDx mode bit is 0. 1 Link state is driven on LEDx_L [0] if the corresponding LEDx mode bit is 0. The reset value is determined by the value of PR_AD[3] during the deasserting edge of RST_L or LRST_L. When software writes this bit with 1, the LED Hardware Mode bit is set. See Section 3.19.2 for a description of the LED operation in this state.
31	LEDHSC	R/W1TC	H/W	LED Hardware State Clear. Selects either link or differential pair state to be reflected by the LEDs. When read, this bit reflects the value of the LED H/W State. When the LED Hardware State is: 0 Differential pair receiver state is driven on all eight LED signals if the corresponding LEDx mode bit is 0. 1 Link state is driven on LEDx_L [0] if the corresponding LEDx mode bit is 0. The reset value is determined by the value of PR_AD[3] during the deasserting edge of RST_L or LRST_L. When software writes this bit with a 1, the LED Hardware Mode bit is cleared. See Section 3.19.2 for a description of the LED operation in this state.

4.6.1.3 Scratchpad Registers

These registers provide read/write state for software and are not associated with any other chip functionality.

Ch. 255 Byte Offset	4010h:402Fh
Size	32 bytes

Each byte is as follows:

Bit	Name	Access	Reset Value	Description
7:0	SCRATCH	R/W	0	Byte-accessible scratchpad state, used for reading and writing application-specific software messages/information.

4.6.1.4 PCI Doorbell Registers

These registers provide local software control for assertion of interrupt INTA_L. If one or more of the interrupt bits is set and the corresponding mask bits is clear, INTA_L is asserted or remains asserted.

4.6.1.4.1 Clear IRQ

Ch. 255 Byte Offset	4054h:4057h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	CLR_IRQ	R/W1TC	0	Controls the clearing of doorbell interrupt bits. Writing 1 to a bit clears it. Writing 0 to a bit has no effect. When read, this register returns the state of the interrupt bits.

4.6.1.4.2 Clear IRQ Mask

Ch. 255 Byte Offset	4058h:405Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	CLR_IRQM	R/W1TC	1	Controls the clearing of doorbell interrupt mask bits. Writing 1 to a bit clears it. Writing 0 to a bit has no effect. When read, this register returns the state of the interrupt mask bits.

4.6.1.4.3 Set IRQ

Ch. 255 Byte Offset	405Ch:405Fh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	SET_IRQ	R/W1TS	0	Controls the setting of doorbell interrupt bits. Writing 1 to a bit sets it. Writing 0 to a bit has no effect. When read, this register returns the state of the interrupt bits.

4.6.1.4.4 Set IRQ Mask

Ch. 255 Byte Offset	4060h:4063h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	SET_IRQM	R/W1TS	1	Controls the setting of doorbell interrupt mask bits. Writing 1 to a bit sets it. Writing 0 to a bit has no effect. When read, this register returns the state of the interrupt mask bits.

4.6.2 Port Map Table

These registers hold the PCI configuration, memory, and I/O address regions and PCI control for SG2010's link partners, and are used to determine the output port for address-routed frames directed into the fabric.

4.6.2.1 Port Map Command and Bridge Control

Port 0 Ch. 255 Byte Offset	4100h:4103h
Port 1 Ch. 255 Byte Offset	4120h:4123h
Size	8 bytes total, 4 bytes per port

Bit	Name	Access	Reset Value	Description
15:0	CMDP*x*	R/W	0	Port Map *x* Command register. Contains a copy of the Command configuration register for the link partner connected to port *x*.
31:16	BCP*x*	R/W	0	Port Map *x* Bridge Control register. Contains a copy of the Bridge Control configuration register for the link partner connected to port *x*.

4.6.2.2 Port Map Bus Numbers

The Port Map Bus Numbers contain the PCI bus numbers defining the configuration ranges for port 0 and port 1.

Port 0 Ch. 255 Byte Offset	4104h:4107h
Port 1 Ch. 255 Byte Offset	4124h:4127h
Size	8 bytes total, 4 bytes per port

Bit	Name	Access	Reset Value	Description
7:0	PRBUS*x*	R/W	0	Port Map *x* Primary Bus Number. Contains a copy of the Primary Bus Number register for the link partner connected to port *x*.
15:8	SECBUS*x*	R/W	0	Port Map *x* Secondary Bus Number. Contains a copy of the Secondary Bus Number register for the link partner connected to port *x*.
23:16	SUBBUS*x*	R/W	0	Port Map *x* Subordinate Bus Number. Contains a copy of the Subordinate Bus Number register for the link partner connected to port *x*.
31:24	RES	R	0	Reserved

4.6.2.3 Port Map I/O Base and Limit

The Port Map I/O Base and Limit contains the lower I/O base and limit bits for port 0 and port 1.

Port 0 Ch. 255 Byte Offset	4108h:410Bh
Port 1 Ch. 255 Byte Offset	4128h:412Bh
Size	8 bytes total, 4 bytes per port

Bit	Name	Access	Reset Value	Description
3:0	RES	R	0	Reserved
7:4	PIOB*x*	R/W	0	Port Map *x* I/O Base. Contains a copy of the I/O Base register for the link partner connected to port *x*. This field holds address bits [15:12].
11:8	RES	R	0	Reserved
15:12	PIOL*x*	R/W	0	Port Map *x* I/O Limit. Contains a copy of the I/O Limit register for the link partner connected to port *x*. This field holds address bits [15:12].
31:16	RES	R	0	Reserved.

4.6.2.4 Port Map Memory Base and Limit

The Port Map Memory Base and Limit define the memory address ranges for port 0 and port 1.

Port 0 Ch. 255 Byte Offset	410Ch:410Fh
Port 1 Ch. 255 Byte Offset	412Ch:412Fh
Size	8 bytes total, 4 bytes per port

Bit	Name	Access	Reset Value	Description
3:0	RES	R	0	Reserved
15:4	PMB*x*	R/W	0	Port Map *x* Memory Base. Contains a copy of the Memory Base register for the link partner connected to port *x*. This field holds address bits [31:20].
19:16	RES	R	0	Reserved
31:20	PML*x*	R/W	0	Port Map *x* Memory Limit. Contains a copy of the Memory Limit register for the link partner connected to port *x*. This field holds address bits [31:20].

4.6.2.5 Port Map Prefetchable Memory Base and Limit

The Port Map Prefetchable Memory Base and Limit define the lower bits of the prefetchable memory address ranges for port 0 and port 1.

Port 0 Ch. 255 Byte Offset	4110h:4113h
Port 1 Ch. 255 Byte Offset	4130h:4133h
Size	8 bytes total, 4 bytes per port

Bit	Name	Access	Reset Value	Description
3:0	RES	R	0	Reserved
15:4	PPMB*x*	R/W	0	Port Map *x* Prefetchable Memory Base. Contains a copy of the Prefetchable Memory Base register for the link partner connected to port *x*. This field holds address bits [31:20].
19:16	RES	R	0	Reserved
31:20	PPML*x*	R/W	0	Port Map *x* Prefetchable Memory Limit. Contains a copy of the Prefetchable Memory Limit register for the link partner connected to port *x*. This field holds address bits [31:20].

4.6.2.6 Port Map Prefetchable Memory Base Upper 32 Bits

The Port Map Prefetchable Memory Base and Limit define the upper bits of the prefetchable memory address ranges for port 0 and port 1.

Port 0 Ch. 255 Byte Offset	4114h:4117h
Port 1 Ch. 255 Byte Offset	4134h:4137h
Size	8 bytes total, 4 bytes per port

Bit	Name	Access	Reset Value	Description
31:0	PPMBU*x*	R/W	0	Port Map *x* Prefetchable Memory Base Upper 32 Bits. Contains a copy of the Prefetchable Memory Base Upper 32 Bits register for the link partner connected to port *x*. This field holds address bits [63:32].

4.6.2.7 Port Map Prefetchable Memory Limit Upper 32 Bits

The Port Map Prefetchable Memory Base and Limit define the upper bits of the prefetchable memory address ranges for port 0 and port 1.

Port 0 Ch. 255 Byte Offset	4118h:411Bh
Port 1 Ch. 255 Byte Offset	4138h:413Bh
Size	8 bytes total, 4 bytes per port

Bit	Name	Access	Reset Value	Description
31:0	PPMLU*x*	R/W	0	Port Map *x* Prefetchable Memory Limit Upper 32 Bits. Contains a copy of the Prefetchable Memory Limit Upper 32 Bits register for the link partner connected to port *x*.This field holds address bits [63:32].

4.6.2.8 Port Map I/O Base and Limit Upper 16 Bits

The Port Map I/O Base and Limit Upper 16 Bits contains the upper I/O base and limit bits for port 0 and port 1.

Port 0 Ch. 255 Byte Offset	411Ch:411Fh
Port 1 Ch. 255 Byte Offset	413Ch:413Fh
Size	8 bytes total, 4 bytes per port

Bit	Name	Access	Reset Value	Description
15:0	PIOBU*x*	R/W	0	Port Map *x* I/O Base Upper 16 bits. Contains a copy of the I/O Base Upper 16 Bits register for the link partner connected to port *x*. This field holds address bits [31:16].
31:16	PIOLU*x*	R/W	0	Port Map *x* I/O Limit Upper 16 bits. Contains a copy of the I/O Limit Upper 16 Bits register for the link partner connected to port *x*. This field holds address bits [31:16].

4.6.3 Multicast Registers

Multicast Group 0 Ch. 255 Byte Offset	4200h:4203h
Multicast Group 1 Ch. 255 Byte Offset	4204h:4207h
Multicast Group 2 Ch. 255 Byte Offset	4208h:420Bh
Multicast Group 3 Ch. 255 Byte Offset	420Ch:420Fh
Multicast Group 4 Ch. 255 Byte Offset	4210h:4213h
Multicast Group 5 Ch. 255 Byte Offset	4214h:4217h
Multicast Group 6 Ch. 255 Byte Offset	4218h:421Bh
Multicast Group 7 Ch. 255 Byte Offset	421Ch:421Fh
Multicast Group 8 Ch. 255 Byte Offset	4220h:4223h
Multicast Group 9 Ch. 255 Byte Offset	4224h:4227h
Multicast Group 10 Ch. 255 Byte Offset	4228h:422Bh
Multicast Group 11 Ch. 255 Byte Offset	422Ch:422Fh
Multicast Group 12 Ch. 255 Byte Offset	4230h:4233h
Multicast Group 13 Ch. 255 Byte Offset	4234h:4237h
Multicast Group 14 Ch. 255 Byte Offset	4238h:423Bh
Multicast Group 15 Ch. 255 Byte Offset	423Ch:423Fh
Multicast Group 16 Ch. 255 Byte Offset	4240h:4243h
Multicast Group 17 Ch. 255 Byte Offset	4244h:4247h
Multicast Group 18 Ch. 255 Byte Offset	4248h:424Bh
Multicast Group 19 Ch. 255 Byte Offset	424Ch:424Fh
Multicast Group 20 Ch. 255 Byte Offset	4250h:4253h
Multicast Group 21 Ch. 255 Byte Offset	4254h:4257h
Multicast Group 22 Ch. 255 Byte Offset	4258h:425Bh
Multicast Group 23 Ch. 255 Byte Offset	425Ch:425Fh
Multicast Group 24 Ch. 255 Byte Offset	4260h:4263h
Multicast Group 25 Ch. 255 Byte Offset	4264h:4267h
Multicast Group 26 Ch. 255 Byte Offset	4268h:426Bh
Multicast Group 27 Ch. 255 Byte Offset	426Ch:426Fh
Multicast Group 28 Ch. 255 Byte Offset	4270h:4273h
Multicast Group 29 Ch. 255 Byte Offset	4274h:4277h
Multicast Group 30 Ch. 255 Byte Offset	4278h:427Bh
Multicast Group 31 Ch. 255 Byte Offset	427Ch:427Fh
Size	1 Dword per group, 32 Dwords total

Bit	Name	Access	Reset Value	Description
1:0	MG*x*PORT	R/W	0	These bits identify the output ports that belong to multicast group *x*. Bit 0 corresponds to Port0 and bit 1 correspond to Port1. When a bit is: 0 The corresponding port is not a multicast group member. 1 The corresponding output port is a multicast group member. Programmed by software.
15:2	RES	R	0	Reserved
17:16	MG*x*ACK	R	0	These bits reflect the write acknowledge frames received in response to a multicast frame sent for Multicast Group *x*. Bit 0 corresponds to Port0 and bit 1 correspond to Port1. When a bit is: 0 A multicast write acknowledge has not been received on the corresponding port for this multicast group. 1 A multicast write acknowledge with the Final Multicast Ack bit set has been received on the corresponding port for this multicast group. The SG2010 clears these bits when all expected multicast write acknowledges are received or when software writes the Clear Multicast Ack bit.
24:18	RES	R	0	Reserved
25	NTS	R		Multicast nack tracking support. Reads as 0 to indicate that multicast nack tracking is not supported.
26	NSL	R		Multicast nack select. Reads as 0 since multicast nack tracking is not supported.
27	CLRACK	R/W1TC	0	Clear Multicast Ack. Writing 1 to this bit clears all multicast acknowledge bits in this group. Writing 0 has no effect. When read, this bit always returns 0.
28	MGINPUT	R/W	0	Returns the input port for this multicast group. The bit is used by switches and is not used by the SG2010, but can be written and read by software.
31:29	RES	R	0	Reserved

4.6.4 Semaphore Registers

The SG2010 implements eight semaphores. Each semaphore has eight operations. Each semaphore operation is performed by reading the corresponding semaphore register Dword. Write data is discarded and have no effect.

If a semaphore Dword location is read and all of the byte enables for that Dword are disabled, the semaphore operation is not performed.

4.6.4.1 Semaphore N Clear

Semaphore 0 Ch. 255 Byte Offset	4400h:4403h
Semaphore 1 Ch. 255 Byte Offset	4420h:4423h
Semaphore 2 Ch. 255 Byte Offset	4440h:4443h
Semaphore 3 Ch. 255 Byte Offset	4460h:4463h
Semaphore 4 Ch. 255 Byte Offset	4480h:4483h
Semaphore 5 Ch. 255 Byte Offset	44A0h:44A3h
Semaphore 6 Ch. 255 Byte Offset	44C0h:44C3h
Semaphore 7 Ch. 255 Byte Offset	44E0h:44E3h
Size	32 bytes total, 4 bytes per semaphore

Bit	Name	Access	Reset Value	Description
7:0	SEM*x*_CLR	RTC	0	When this register is read, the SG2010 returns the value of Semaphore *N*, and then clears the value of Semaphore *N* to 0.
31:8	RES	R	0	The semaphore is eight bits wide and this field always reads as zero.

4.6.4.2 Semaphore N Set

Semaphore 0 Ch. 255 Byte Offset	4404h:4407h
Semaphore 1 Ch. 255 Byte Offset	4424h:4427h
Semaphore 2 Ch. 255 Byte Offset	4444h:4447h
Semaphore 3 Ch. 255 Byte Offset	4464h:4467h
Semaphore 4 Ch. 255 Byte Offset	4484h:4487h
Semaphore 5 Ch. 255 Byte Offset	44A4h:44A7h
Semaphore 6 Ch. 255 Byte Offset	44C4h:44C7h
Semaphore 7 Ch. 255 Byte Offset	44E4h:44E7h
Size	32 bytes total, 4 bytes per semaphore

Bit	Name	Access	Reset Value	Description
7:0	SEM*x*_SET	RTS	0	When this register is read, the SG2010 returns the value of Semaphore *N*, and then sets the value of Semaphore *N* to FFh.
31:8	RES	R	0	The semaphore is eight bits wide and this field always reads as zero.

4.6.4.3 Semaphore N Decrement

Semaphore 0 Ch. 255 Byte Offset	4408h:440Bh
Semaphore 1 Ch. 255 Byte Offset	4428h:442Bh
Semaphore 2 Ch. 255 Byte Offset	4448h:444Bh
Semaphore 3 Ch. 255 Byte Offset	4468h:446Bh
Semaphore 4 Ch. 255 Byte Offset	4488h:448Bh
Semaphore 5 Ch. 255 Byte Offset	44A8h:44ABh

Semaphore 6 Ch. 255 Byte Offset	44C8h:44CBh
Semaphore 7 Ch. 255 Byte Offset	44E8h:44EBh
Size	32 bytes total, 4 bytes per semaphore

Bit	Name	Access	Reset Value	Description
7:0	SEM*x*_DEC	RTDEC	0	When this register is read, the SG2010 returns the value of Semaphore *N*, and then decrements the value of Semaphore *N*. The decrement operation sticks at 0.
31:8	RES	R	0	The semaphore is eight bits wide and this field always reads as zero.

4.6.4.4 Semaphore N Increment

Semaphore 0 Ch. 255 Byte Offset	440Ch:440Fh
Semaphore 1 Ch. 255 Byte Offset	442Ch:442Fh
Semaphore 2 Ch. 255 Byte Offset	444Ch:444Fh
Semaphore 3 Ch. 255 Byte Offset	446Ch:446Fh
Semaphore 4 Ch. 255 Byte Offset	448Ch:448Fh
Semaphore 5 Ch. 255 Byte Offset	44ACh:44AFh
Semaphore 6 Ch. 255 Byte Offset	44CCh:44CFh
Semaphore 7 Ch. 255 Byte Offset	44ECh:44EFh
Size	32 bytes total, 4 bytes per semaphore

Bit	Name	Access	Reset Value	Description
7:0	SEM*x*_INC	RTINC	0	When this register is read, the SG2010 returns the value of Semaphore *N*, and then increments the value of Semaphore *N*. The increment operation sticks at FFh.
31:8	RES	R	0	The semaphore is eight bits wide and this field always reads as zero.

4.6.4.5 Semaphore N Reserved 0

Semaphore 0 Ch. 255 Byte Offset	4410h:4413h
Semaphore 1 Ch. 255 Byte Offset	4430h:4433h
Semaphore 2 Ch. 255 Byte Offset	4450h:4453h
Semaphore 3 Ch. 255 Byte Offset	4470h:4473h
Semaphore 4 Ch. 255 Byte Offset	4490h:4493h
Semaphore 5 Ch. 255 Byte Offset	44B0h:44B3h
Semaphore 6 Ch. 255 Byte Offset	44D0h:44D3h
Semaphore 7 Ch. 255 Byte Offset	44F0h:44F3h
Size	32 bytes total, 4 bytes per semaphore

Bit	Name	Access	Reset Value	Description
7:0	SEM*x*_RES0	R	0	When this register is read, the value of the semaphore is returned. No semaphore operation is performed.
31:8	RES	R	0	The semaphore is eight bits wide and this field always reads as zero.

4.6.4.6 Semaphore N Increment if 0

Semaphore 0 Ch. 255 Byte Offset	4414h:4417h
Semaphore 1 Ch. 255 Byte Offset	4434h:4437h
Semaphore 2 Ch. 255 Byte Offset	4454h:4457h
Semaphore 3 Ch. 255 Byte Offset	4474h:4477h
Semaphore 4 Ch. 255 Byte Offset	4494h:4497h
Semaphore 5 Ch. 255 Byte Offset	44B4h:44B7h
Semaphore 6 Ch. 255 Byte Offset	44D4h:44D7h
Semaphore 7 Ch. 255 Byte Offset	44F4h:44F7h
Size	32 bytes total, 4 bytes per semaphore

Bit	Name	Access	Reset Value	Description
7:0	SEM*x*_INC0	RTINC	0	When this register is read, the SG2010 returns the value of Semaphore *N*, and then increments the value of Semaphore *N* if the previous value was 0.
31:8	RES	R	0	The semaphore is eight bits wide and this field always reads as zero.

4.6.4.7 Semaphore N Reserved 1

Semaphore 0 Ch. 255 Byte Offset	4418h:441Bh
Semaphore 1 Ch. 255 Byte Offset	4438h:443Bh
Semaphore 2 Ch. 255 Byte Offset	4458h:445Bh
Semaphore 3 Ch. 255 Byte Offset	4478h:447Bh
Semaphore 4 Ch. 255 Byte Offset	4498h:449Bh
Semaphore 5 Ch. 255 Byte Offset	44B8h:44BBh
Semaphore 6 Ch. 255 Byte Offset	44D8h:44DBh
Semaphore 7 Ch. 255 Byte Offset	44F8h:44FBh
Size	32 bytes total, 4 bytes per semaphore

Bit	Name	Access	Reset Value	Description
7:0	SEM*x*_RES1	R	0	When this register is read, the value of the semaphore is returned. No semaphore operation is performed.
31:8	RES	R	0	The semaphore is eight bits wide and this field always reads as zero.

4.6.4.8 Semaphore N Increment if not 0

Semaphore 0 Ch. 255 Byte Offset	441Ch:441Fh
Semaphore 1 Ch. 255 Byte Offset	443Ch:443Fh
Semaphore 2 Ch. 255 Byte Offset	445Ch:445Fh
Semaphore 3 Ch. 255 Byte Offset	447Ch:447Fh
Semaphore 4 Ch. 255 Byte Offset	449Ch:449Fh
Semaphore 5 Ch. 255 Byte Offset	44BCh:44BFh

Semaphore 6 Ch. 255 Byte Offset	44DCh:44DFh
Semaphore 7 Ch. 255 Byte Offset	44FCh:44FFh
Size	32 bytes total, 4 bytes per semaphore

Bit	Name	Access	Reset Value	Description
7:0	SEM*x*_INCN0	RTINC	0	When this register is read, the SG2010 returns the value of Semaphore *N*, and then increments the value of Semaphore *N* if the previous value was not 0. The increment operation sticks at FFh.
31:8	RES	R	0	The semaphore is eight bits wide and this field always reads as zero.

4.6.5 Register Path Protection

There are four entries in the Register Path Protection Table. Any incoming frame used for register access is compared against all enabled paths. If all enabled path comparisons fail, the register access is not performed and an event is signaled. Otherwise, the register access is allowed.

Path 0 Ch. 255 Byte Offset	4F00:4F03
Path 1 Ch. 255 Byte Offset	4F04:4F07
Path 2 Ch. 255 Byte Offset	4F08:4F0B
Path 3 Ch. 255 Byte Offset	4F0C:4F0F
Size	4 bytes per entry, 16 bytes total

Bit	Name	Access	Reset Value	Description
20:0	RPP*x*	R/W	0	This field contains a transformed (inverted and reversed) path specification used for path protection checks against incoming frames accessing any SG2010 register in any address space. The transformed path is the path from the SG2010 to the origin.
21	INPUT	R/W	0	This field contains the input port that accompanies the above path specification. The input port used by the frame must match the input port specified in this field.
30:22	RES	R	0	Reserved
31	PPEN*x*	R/W	0	Enables path protection using the path in bits [20:0]. If 0, path protection is not performed against this register. If 1, path protection is performed against this register.

4.6.6 Port State Table

4.6.6.1 Port State *x* Link Map

Port 0 Ch. 255 Byte Offset	4F20:4F23
Port 1 Ch. 255 Byte Offset	4F28:4F2B
Size	1 Dword per port, 2 Dwords total

Bit	Name	Access	Reset Value	Description
1:0	P2LMAP*x*	R	0	These bits reflect the links that belong to Port *x*. A one in bit [0] indicates that Link 0 belongs to Port *x*. A one in bit [1] indicates that Link 1 belongs to Port *x*. A 0 indicates that the link does not belong to that port. Set by the SG2010 during fabric enumeration.
31:2	RES	R	0	Reserved

4.6.6.2 Port State *x* Control and Status

Port 0 Ch. 255 Byte Offset	4F24:4F27
Port 1 Ch. 255 Byte Offset	4F2C:4F2F
Size	1 Dword per port, 2 Dwords total

Bit	Name	Access	Reset Value	Description
0	PORTUP	R	1	Port Up/Down State. This bit reflects the state of the Traffic Enable bits for the port's links. When: 0 The port is down (the Traffic Enable bits for all links in the port are clear). 1 The port is up (the Traffic Enable bit is set for at least one link in the port).
1	RES	R	0	Reserved.
2	ROOT	R	0	When: 0 This port is not the root port. 1 This port is the root port (on the path to the root).
3	RES	R	0	Reserved
4	PMTENA	R/W	0	Port Map Table Enable. When: 0 The Port Map Table is disabled for address routing, and therefore no positively decoded address-routed frames can exit this port. 1 The Port Map Table is enabled for address decoding for address-routed frames. After fabric enumeration, the downstream ports in the PCI hierarchy are enabled by hardware. This bit may also be enabled or disabled by software.

5	SARENA	R/W	0	Smart Address Enable. When the PMTENA bit is set and this bit is: 0 All address types are decoded. 1 Disables decoding of configuration addresses against the Port Map Table. If the Port Map Table Enable bit is not set, this bit has no effect on address decoding.
6	RESETDIS	R/W	0	Maskable Reset Disable. When: 0 The SG2010 propagates a maskable reset received on the corresponding port and then resets. 1 The SG2010 ignores maskable resets received on the corresponding port. This bit is written during fabric enumeration and can also be modified by software.
31:7	RES	R	0	Reserved.

4.6.7 Link State Table

These registers contain link-related control and status information. Each register is instantiated once per link.

4.6.7.1 Link Control and Status Register

Link 0 Ch. 255 Byte Offset	4F80h:4F83h
Link 1 Ch. 255 Byte Offset	4FC0h:4FC3h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
3:0	PORTNUM	R	H/W	Specifies the Port State Table and, if used, the Port Map Table set used by this link. Valid values are either 0 or 1. Initialized by hardware during fabric enumeration. This is based on the Port State Link Map. Table Not meaningful if the link is down.
4	LDIS	R/W	0	Link Disable. When written with 1, causes the SG2010 to disable this link's transmitters and set the Link Down event bit. When written with 0, the SG2010 attempts to synchronize the link, and, if successful, sets the Link Up event bit.
5	ROOT	R	0	When: 0 The port that this link belongs to is a non-root port. 1 The port that this link belongs to is a root port.
6	F8B10B	WRZ	0	Force 8B/10B error. When written with 1, the SG2010 forces one 8B/10B error to occur on the link. This register always returns 0 when read.
7	FCRC	WRZ	0	Force CRC error. When written with 1, the SG2010 forces one CRC error to occur on the link. This register always returns 0 when read.

Bit	Name	Access	Reset Value	Description
9:8	LINKST	R	H/W	Current link state. This field reflects the current synchronization state of the link, where: 11b Linked state (fully synchronized) 10b Reply state 01b Acknowledge state 00b Call state (Not synchronized)
11:10	HLINKST	R/W1TC	0	Highest link state. Reflects the highest link state transmitted by the link since either a reset or a software clear, where: 11b Linked state (fully synchronized) 10b Reply state 01b Acknowledge state 00b Call state (Not synchronized) Software can clear this state by writing a 1. A write of 0 has no effect.
13:12	RLINKST	R	H/W	Current link state of the receiver (link partner). This field reflects the current synchronization state of the link from the receiver point of view, where: 11b Linked state (fully synchronized) 10b Reply state 01b Acknowledge state 00b Call state (Not synchronized)
15:14	RHLINKST	R/W1TC	0	Highest link state. Reflects the highest link state received by the link since either a reset or a software clear, where: 11b Linked state (fully synchronized) 10b Reply state 01b Acknowledge state 00b Call state (Not synchronized) Software can clear this state by writing a 1. A write of 0 has no effect.
19:16	COMTYP	R	0	Attached component type. Assigned during fabric enumeration. Currently assigned values: Bit [16]: When 1, has edge node functionality Bit [17]: When 1, has switch functionality Bits [19:18]: Reserved
23:20	RES	R	0	Reserved
24	TENA	R/W	1	Traffic Enable. The product of this bit and Link Partner Traffic Enable enables credit-based frames to be sent on this link. Set by hardware after the chip is reset, or when a Set State frame is received with the Set_TEN bit set. Cleared by hardware as described in Section 3.9.1.2. Software may set or clear this bit.

Bit	Name	Access	Reset Value	Description
27:25	LNKSPD	R	000b	Indicates the link speed of this link, and dictates the reset value of the Bandwidth Count register. Reads as 000b to indicate that the full speed of this link is 2 Gbit/second in each direction.
31:25	RES	R	0	Reserved

4.6.7.2 Link Partner Fabric ID

This register contains the Fabric ID (FID) and other information retrieved from the last I Am special frame received from the link partner.

Link 0 Ch. 255 Byte Offset	4F84h:4F87h
Link 1 Ch. 255 Byte Offset	4FC4h:4FC7h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
2:0	RES	R	0	Reserved
3	LP_TEN	R	1	Link Partner Traffic Enable. Contains the inverse of the TDIS bit received in the last I Am special frame
4	RES	R	0	Reserved
7:5	LPPFN	R	111b	Contains the parallel fabric number (PFN) of the FID of the link partner, received in the last I Am special frame
10:8	LPTC	R	111b	Contains the turn count of the FID of the link partner, received in the last I Am special frame.
13:11	LPT0	R	111b	Contains turn 0 of the FID of the link partner, received in the last I Am special frame.
16:14	LPT1	R	111b	Contains turn 1 of the FID of the link partner, received in the last I Am special frame.
19:17	LPT2	R	111b	Contains turn 2 of the FID of the link partner, received in the last I Am special frame.
22:20	LPT3	R	111b	Contains turn 3 of the FID of the link partner, received in the last I Am special frame.
25:23	LPT4	R	111b	Contains turn 4 of the FID of the link partner, received in the last I Am special frame.
28:26	LPT5	R	111b	Contains turn 5 of the FID of the link partner, received in the last I Am special frame.
31:29	LPT6	R	111b	Contains turn 6 of the FID of the link partner, received in the last I Am special frame.

4.6.7.3 8B/10B and CRC Error Count

Link 0 Ch. 255 Byte Offset	4F88h:4F8Bh
Link 1 Ch. 255 Byte Offset	4FC8h:4FCBh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
15:0	CRC	R/W	0	The CRC counter increments by 1 every time a CRC error is detected by this link. After the counter reaches its maximum value, it wraps and starts counting from 0. A CRC Counter Wrap event is signaled on overflow.
31:16	8B10B	R/W	0	The 8B/10B counter increments by 1 every time an 8B/10B error is detected by this link. After the counter reaches its maximum value, it wraps and starts counting from 0. An 8B/10B Counter Wrap event is signaled on overflow.

4.6.7.4 Frame Counter

Link 0 Ch. 255 Byte Offset	4F8Ch:4F8Fh
Link 1 Ch. 255 Byte Offset	4FCCh:4FCFh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	FRAMEC	R/W	0	The frame counter increments by 1 every time a non-empty frame is transmitted by this link. After the counter reaches its maximum value, it wraps and starts counting from 0. A Frame Counter Wrap event is signaled on overflow.

4.6.7.5 Line Counter

Link 0 Ch. 255 Byte Offset	4F90h:4F93h
Link 1 Ch. 255 Byte Offset	4FD0h:4FD3h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	LINEC	R/W	0	The line counter increments by 1 every time a 128-bit non-empty line is transmitted by this link. After the counter reaches its maximum value, it wraps and starts counting from 0. A Line Counter Wrap event is signaled on overflow.

4.6.7.6 Empty Frame Counter

Link 0 Ch. 255 Byte Offset	4F94h:4F97h
Link 1 Ch. 255 Byte Offset	4FD4h:4FD7h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ELINEC	R/W	0	The empty frame counter increments by 1 every time an empty frame is transmitted by this link. After the counter reaches its maximum value, it wraps and starts counting from 0. An Empty Frame Counter Wrap event is signaled on overflow.

4.6.7.7 Asynchronous Write Credit Count

Link 0 Ch. 255 Byte Offset	4F98h
Link 1 Ch. 255 Byte Offset	4FD8h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	AWCC	R	0	This register reflects the current value of this link's counter for asynchronous write credits reserving buffer space in the link partner. The SG2010 does not implement this counter and it reads as 0. Asynchronous credits are aliased with address-routed credits.

4.6.7.8 Isochronous Write Credit Count

Link 0 Ch. 255 Byte Offset	4F99h
Link 1 Ch. 255 Byte Offset	4FD9h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	IWCC	R	0	This register reflects the current value of this link's counter for isochronous write credits reserving buffer space in the link partner. This counter is reset when the link goes down.

4.6.7.9 HP-Asynchronous Write Credit Count

Link 0 Ch. 255 Byte Offset	4F9Ah
Link 1 Ch. 255 Byte Offset	4FDAh
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	HPWCC	R	0	This register reflects the current value of this link's counter for the HP-Asynchronous write credits reserving buffer space in the link partner. The SG2010 does not implement this counter and it reads as 0. HP-asynchronous credits are aliased with provisioning credits.

4.6.7.10 Multicast Write Credit Count

Link 0 Ch. 255 Byte Offset	4F9Bh
Link 1 Ch. 255 Byte Offset	4FDBh
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	MCCC	R	0	This register reflects the current value of this link's counter for multicast credits reserving buffer space in the link partner. This counter is reset when the link goes down.

4.6.7.11 Address-Routed Write Credit Count

Link 0 Ch. 255 Byte Offset	4F9Ch
Link 1 Ch. 255 Byte Offset	4FDCh
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	ARWCC	R	0	This register reflects the current value of this link's counter for address-routed write credits reserving buffer space in the link partner. This counter is reset when the link goes down.

4.6.7.12 HP-Isochronous Write Credit Count

Link 0 Ch. 255 Byte Offset	4F9Dh
Link 1 Ch. 255 Byte Offset	4FDDh
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	HIWCC	R	0	This register reflects the current value of this link's counter for HP-isochronous write credits reserving buffer space in the link partner. The SG2010 does not implement this counter and it reads as 0. HP-isochronous credits are aliased with isochronous credits.

4.6.7.13 Provisioning Write Credit Count

Link 0 Ch. 255 Byte Offset	4F9Eh
Link 1 Ch. 255 Byte Offset	4FDEh
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	PWCC	R	0	This register reflects the current value of this link's counter for provisioning/HP-asynchronous write credits reserving buffer space in the link partner. This counter is reset when the link goes down.

4.6.7.14 Turn *N* Write Credit Count

Link 0 Turn 0 Ch. 255 Byte Offset	4FA0h
Link 0 Turn 1 Ch. 255 Byte Offset	4FA1h
Link 0 Turn 2 Ch. 255 Byte Offset	4FA2h
Link 0 Turn 3 Ch. 255 Byte Offset	4FA3h
Link 0 Turn 4 Ch. 255 Byte Offset	4FA4h
Link 0 Turn 5 Ch. 255 Byte Offset	4FA5h
Link 0 Turn 6 Ch. 255 Byte Offset	4FA6h
Link 0 Turn 7 Ch. 255 Byte Offset	4FA7h
Link 1 Turn 0 Ch. 255 Byte Offset	4FE0h
Link 1 Turn 1 Ch. 255 Byte Offset	4FE1h
Link 1 Turn 2 Ch. 255 Byte Offset	4FE2h
Link 1 Turn 3 Ch. 255 Byte Offset	4FE3h
Link 1 Turn 4 Ch. 255 Byte Offset	4FE4h
Link 1 Turn 5 Ch. 255 Byte Offset	4FE5h
Link 1 Turn 6 Ch. 255 Byte Offset	4FE6h
Link 1 Turn 7 Ch. 255 Byte Offset	4FE7h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	NT*n*WCC	R	0	This register reflects the current value of this link’s counter for turn write credits reserving buffer space in the link partner. There are eight turn credit counters per link. This counter is reset when the link goes down.

4.6.7.15 Asynchronous Request/Isochronous Request Credit Count

Link 0 Ch. 255 Byte Offset	4FA8h
Link 1 Ch. 255 Byte Offset	4FE8h
Size	1 byte

Bit	Name	Access	Reset Value	Description
3:0	ARCC	R	0	This register reflects the current value of this link’s counter for asynchronous request credits reserving buffer space in the link partner. The SG2010 does not implement this counter and it reads as 0. Asynchronous credits are aliased with address-routed credits.
7:4	IRCC	R	0	This register reflects the current value of this link’s counter for isochronous request credits reserving buffer space in the link partner. This counter is reset when the link goes down.

4.6.7.16 HP-Asynchronous Request Credit Count

Link 0 Ch. 255 Byte Offset	4FA9h
Link 1 Ch. 255 Byte Offset	4FE9h
Size	1 byte

Bit	Name	Access	Reset Value	Description
3:0	HPRCC	R	0	This register reflects the current value of this link's counter for the HP-Asynchronous request credits reserving buffer space in the link partner. The SG2010 does not implement this counter and it reads as 0. HP-asynchronous credits are aliased with provisioning credits.
7:4	RES	R	0	Reserved

4.6.7.17 Address-Routed/HP-Isochronous Request Credit Count

Link 0 Ch. 255 Byte Offset	4FAAh
Link 1 Ch. 255 Byte Offset	4FEAh
Size	1 byte

Bit	Name	Access	Reset Value	Description
3:0	ARRCC	R	0	This register reflects the current value of this link's counter for address-routed request credits reserving buffer space in the link partner. This counter is reset when the link goes down.
7:4	HIRCC	R	0	This register reflects the current value of this link's counter for HP-Isochronous request credits reserving buffer space in the link partner. The SG2010 does not implement this counter and it reads as 0. HP-isochronous credits are aliased with isochronous credits.

4.6.7.18 Provisioning Request Credit Count

Link 0 Ch. 255 Byte Offset	4FABh
Link 1 Ch. 255 Byte Offset	4FEBh
Size	1 byte

Bit	Name	Access	Reset Value	Description
3:0	PRCC	R	0	This register reflects the current value of this link's counter for provisioning request credits reserving buffer space in the link partner. This counter is reset when the link goes down.
7:4	RES	R	0	Reserved

4.6.7.19 Turn *N* Request Credit Count

Link 0 Turn 0/1 Ch. 255 Byte Offset	4FACh
Link 0 Turn 2/3 Ch. 255 Byte Offset	4FADh
Link 0 Turn 4/5 Ch. 255 Byte Offset	4FAEh
Link 0 Turn 6/7 Ch. 255 Byte Offset	4FAFh
Link 1 Turn 0/1 Ch. 255 Byte Offset	4FECh
Link 1 Turn 2/3 Ch. 255 Byte Offset	4FEDh

Link 1 Turn 4/5 Ch. 255 Byte Offset	4FEEh
Link 1 Turn 6/7 Ch. 255 Byte Offset	4FEFh
Size	1 byte

Bit	Name	Access	Reset Value	Description
3:0	NT*n*RCC	R	0	This register reflects the current value of this link's counter for turn *n* request credits reserving buffer space in the link partner. There are eight four-bit turn request credit counters per link. This counter is reset when the link goes down.
7:4	NT*n*RCC	R	0	This register reflects the current value of this link's counter for turn *n+1* request credits reserving buffer space in the link partner. There are eight 4-bit turn request credit counters per link. This counter is reset when the link goes down.

4.6.7.20 Bandwidth Count

Link 0 Ch. 255 Byte Offset	4FB0h:4FB1h
Link 1 Ch. 255 Byte Offset	4FF0h:4FF1h
Size	2 bytes per entry, 4 bytes total

Bit	Name	Access	Reset Value	Description
11:0	BWCNT	R/W	5FFh	This register reflects the current value of the bandwidth counter for this link. Does not affect any functionality in the SG2010. Software can manage this register, but it is not required for proper operation.
15:12	RES	R	0	Reserved

4.6.7.21 Differential Pair State

Link 0 Ch. 255 Byte Offset	4FB2h
Link 1 Ch. 255 Byte Offset	4FF2h
Size	1 byte per entry, 2 bytes total

Bit	Name	Access	Reset Value	Description
3:0	TXST	R	0	This register reflects the operational transmit differential pairs for this link. Each bit corresponds to a differential pair.This state is extracted from the differential pair state in last valid synchronization special frame received (Call, Ack, Reply, or Linked frame). If the link is not receiving valid frames, this state may not be accurate. Bits [3:0] correspond to TX*n*P/TX*n*N[3:0]. When a bit is: 0 A differential pair is not operational. 1 A differential pair is operational.
7:4	RXST	R	0	This register reflects the operational receive differential pairs for this link, that is, which pairs are receiving good 8B/10B data. Each bit corresponds to a differential pair. Bits [3:0] correspond to RX*n*P/RX*n*N[3:0]. When a bit is: 0 A differential pair is not operational. 1 A differential pair is operational.

4.6.7.22 Default CoS Credit

The Default CoS Credit is the number of credits the SG2010 uses when initializing its link partner's credit counters after link synchronization.

Link 0 Ch. 255 Byte Offset	4FB4:4FBDh
Link 1 Ch. 255 Byte Offset	4FF4:4FFDh
Size	10 bytes per entry, 20 bytes total

Bit	Name	Access	Reset Value	Description
3:0	DCARR	R	0h	This field reflects the default value of SG2010's asynchronous request line credits.
10:4	DCARW	R	0h	This field reflects the default value of SG2010's asynchronous write line credits.
14:11	DCIR	R	Ch (12)	This field reflects the default value of SG2010's isochronous request line credits.
21:15	DCIW	R	2Eh (46)	This field reflects the default value of SG2010's isochronous write line credits.
25:22	DCHPR	R	0h	This field reflects the default value of SG2010's HP-asynchronous request line credits.
32:26	DCHPW	R	0h	This field reflects the default value of SG2010's HP-asynchronous write line credits. The Dword bit breakdown is: [31:26] – Dword 0 bits[31:26] [32] – Dword 1 bit[0]
36:33	RES	R	0h	Reserved. The Dword bit breakdown is Dword 1 bits[4:1].
43:37	DCMCST	R	2Eh (46)	This field reflects the default value of SG2010's multicast line credits. The Dword bit breakdown is Dword 1 bits[11:5].
47:44	DCASR	R	Ch (12)	This field reflects the default value of SG2010's address-routed request line credits. The Dword bit breakdown is Dword 1 bits[15:12].
54:48	DCASW	R	44h (68)	This field reflects the default value of SG2010's address-routed write line credits. The Dword bit breakdown is Dword 1 bits[22:16].
58:55	DCRR	R	0h	This field reflects the default value of SG2010's HP-Isochronous request line credits. The Dword bit breakdown is Dword 1 bits[26:23].
65:59	DCRW	R	0h	This field reflects the default value of SG2010's HP-Isochronous write line credits. The Dword bit breakdown is: [63:59] – Dword 1 bits[31:27] [65:64] – Dword 2 bits[1:0]
69:66	DCPRR	R	Ch (12)	This field reflects the default value of SG2010's provisioning request line credits. The Dword bit breakdown is Dword 2 bits[5:2].

Bit	Name	Access	Reset Value	Description
76:70	DCPRW	R	38h (52)	This field reflects the default value of SG2010's provisioning write line credits. The Dword bit breakdown is Dword 2 bits[12:6].
79:77	RES	R	0h	Reserved

4.6.7.23 Default Turn Credit

Link 0 Ch. 255 Byte Offset	4FBEh:4FBFh
Link 1 Ch. 255 Byte Offset	4FFEh:4FFFh
Size	2 bytes per entry, 4bytes total

Bit	Name	Access	Reset Value	Description
7:0	DCTW	R	0	This register reflects the SG2010's default turn write credits. Since the SG2010 does not support default turn credits, this register returns 0.
15:8	DCTR	R	0	This register reflects the SG2010's default turn request credits. Since the SG2010 does not support default turn credits, this register returns 0.

4.6.8 Event Registers

4.6.8.1 Chip Event Table

Note: *Currently only event table entries 4 – 6, 8 – 11, 27 and 28 are implemented. All other entries are reserved.*

Entry 0 Ch. 255 Byte Offset	5000h:5001h
Entry 1 Ch. 255 Byte Offset	5002h:5003h
Entry 2 Ch. 255 Byte Offset	5004h:5005h
Entry 3 Ch. 255 Byte Offset	5006h:5007h
Entry 4 Ch. 255 Byte Offset	5008h:5009h
Entry 5 Ch. 255 Byte Offset	500Ah:500Bh
Entry 6 Ch. 255 Byte Offset	500Ch:500Dh
Entry 7 Ch. 255 Byte Offset	500Eh:500Fh
Entry 8 Ch. 255 Byte Offset	5010h:5011h
Entry 9 Ch. 255 Byte Offset	5012h:5013h
Entry 10 Ch. 255 Byte Offset	5014h:5015h
Entry 11 Ch. 255 Byte Offset	5016h:5017h
Entry 12 Ch. 255 Byte Offset	5018h:5019h
Entry 13 Ch. 255 Byte Offset	501Ah:501Bh
Entry 14 Ch. 255 Byte Offset	501Ch:501Dh
Entry 15 Ch. 255 Byte Offset	501Eh:501Fh
Entry 16 Ch. 255 Byte Offset	5020h:5021h
Entry 17 Ch. 255 Byte Offset	5022h:5023h
Entry 18 Ch. 255 Byte Offset	5024h:5025h
Entry 19 Ch. 255 Byte Offset	5026h:5027h
Entry 20 Ch. 255 Byte Offset	5028h:5029h
Entry 21 Ch. 255 Byte Offset	502Ah:502Bh
Entry 22 Ch. 255 Byte Offset	502Ch:502Dh

Entry 23 Ch. 255 Byte Offset	502Eh:502Fh
Entry 24 Ch. 255 Byte Offset	5030h:5031h
Entry 25 Ch. 255 Byte Offset	5032h:5033h
Entry 26 Ch. 255 Byte Offset	5034h:5035h
Entry 27 Ch. 255 Byte Offset	5036h:5037h
Entry 28 Ch. 255 Byte Offset	5038h:5039h
Entry 29 Ch. 255 Byte Offset	503Ah:503Bh
Entry 30 Ch. 255 Byte Offset	503Ch:503Dh
Entry 31 Ch. 255 Byte Offset	503Eh:503Fh
Size	2 bytes per entry, 32 entries

Bit	Name	Access	Reset Value	Description
1:0	EPIND	R/W	0	Event Path Table Index. Selects one of four entries in the Event Path Table.
5:2	RES	R	0	Reserved
6	ESEND	R/W	0	Event Frame Send Mode. When: 0 List mode is used for dispatching chip events. 1 Polled mode is used for dispatching chip events.
7	ELOCAL	R/W	0	Local destination bit. When: 0 The event handler is on a remote node. An event frame is constructed using the information in the destination table. 1 The event handler on this device is used. The EMU address of the event should be used to select the EMU and operation.
14:8	EMUADR	R/W	0	EMU Address. The seven-bit EMU address used for event frames. Bits [14:9] select the EMU, bit [8] identifies the operation.
15	RES	R	0	Reserved

4.6.8.2 Signal Event Table

Note. *Entry 13 is reserved.*

Entry 0 Ch. 255 Byte Offset	5040h:5041h	INTA_L assertion
Entry 1 Ch. 255 Byte Offset	5042h:5043h	INTA_L deassertion
Entry 2 Ch. 255 Byte Offset	5044h:5045h	INTB_L assertion
Entry 3 Ch. 255 Byte Offset	5046h:5047h	INTB_L deassertion
Entry 4 Ch. 255 Byte Offset	5048h:5049h	INTC_L assertion
Entry 5 Ch. 255 Byte Offset	504Ah:504Bh	INTC_L deassertion
Entry 6 Ch. 255 Byte Offset	504Ch:504Dh	INTD_L assertion
Entry 7 Ch. 255 Byte Offset	504Eh:504Fh	INTD_L deassertion
Entry 8 Ch. 255 Byte Offset	5050h:5051h	PME_L assertion
Entry 9 Ch. 255 Byte Offset	5052h:5053h	PME_L deassertion
Entry 10 Ch. 255 Byte Offset	5054h:5055h	ENUM_L assertion
Entry 11 Ch. 255 Byte Offset	5056h:5057h	ENUM_L deassertion

Entry 12 Ch. 255 Byte Offset	5058h:5059h	SERR_L assertion
Entry 13 Ch. 255 Byte Offset	505Ah:505Bh	RESERVED
Size	2 bytes per entry, 14 entries	

Bit	Name	Access	Reset Value	Description
1:0	EPIND	R/W	0	Event Path Table Index. Selects one of four entries in the Event Path Table.
7:2	RES	R	0	Reserved
14:8	EMUADR	R/W	0	EMU Address. The seven-bit EMU address used for signal event frames. Bits [14:9] select the EMU, bit [8] identifies the operation.
15	RES	R	0	Reserved

4.6.8.3 Event Path Table

Entry 0 Ch. 255 Byte Offset	5080h:5083h
Entry 1 Ch. 255 Byte Offset	5084h:5087h
Entry 2 Ch. 255 Byte Offset	5088h:508Bh
Entry 3 Ch. 255 Byte Offset	508Ch:508Fh
Size	4 bytes per entry, 4 entries

Bit	Name	Access	Reset Value	Description
20:0	EPATH	R/W	0	Path specification for the event frame.
23:21	ECOS	R/W	0	Class of service for the event frame.
24	EOUTPORT	R/W	0	Output port for the event frame.
30:25	RES	R	0	Reserved
31	VALID	R/W	0	Entry valid. When: 0 This entry is not valid and an event frame is not sent for the events using this entry. 1 The entry is valid.

4.6.8.4 Event Mask

4.6.8.4.1 Event Mask W1TC

Event Mask WTC 0 Ch. 255 Byte Offset	50A0h:50A3h
Event Mask WTC 1 Ch. 255 Byte Offset	50A4h:50A7h
Size	8 bytes

Bit	Name	Access	Reset Value	Description
31:0	EMSKC	R/W1TC	FFFF FFFFh	Event Mask Write-1-to-Clear. Each bit in this register is an event mask for the corresponding bit in the Raw Event Status register. When: 0 The event is not masked. 1 The event is masked. Writing 1 to a bit clears it. When read, this register returns the value of the event mask register.

4.6.8.4.2 Event Status Mask W1TS

Event Mask WTS 0 Ch. 255 Byte Offset	50C0h:50C3h
Event Mask WTS 1 Ch. 255 Byte Offset	50C4h:50C7h
Size	8 bytes

Bit	Name	Access	Reset Value	Description
31:0	EMSKS	R/W1TS	FFFF FFFFh	Event Mask Write-1-to-Set. Each bit in this register is an event mask for the corresponding bit in the Raw Event Status register. When: 0 The event is not masked. 1 The event is masked. Writing 1 to a bit sets it. When read, this register returns the value of the event mask register.

4.6.8.5 Raw Event Status

Raw Event Status 0 Ch. 255 Byte Offset	50E0h:50E3h
Raw Event Status 1 Ch. 255 Byte Offset	50E4h:50E7h
Size	8 bytes

Bit	Name	Access	Reset Value	Description
31:0	RAWST	R/W1TC	0	Raw Event Status bits. Each bit in this register represents a different unmasked event. The SG2010 sets the bit when the event arbiter selects the event from pending event status. Software writes 1 to a bit to clear it. The individual events are described in Table 4–13.

4.6.8.6 Event Status

Event Status 0 Ch. 255 Byte Offset	5100h:5103h
Event Status 1 Ch. 255 Byte Offset	5104h:5107h
Size	8 bytes

Bit	Name	Access	Reset Value	Description
31:0	EDSTEV	R/W1TC	0	Event Status register. Each event is represented with a bit in this register. The SG2010 sets an event bit when the event occurs, the corresponding event mask bit is clear, and a table lookup is performed. Writing 1 to a bit clears it.

4.6.8.7 Event Status Bit Mappings

Table 4–13 shows the event bit mappings for the Raw Event Status register, Event Mask register, and Event Status register.

Table 4–13 Event Status Bit Assignments

Dword	Dword Bit	PEC	SEC	Event Bit	Set Condition
0	0	5	00h	Port 0 Down	All links in Port 0 are down
0	1	5	40h	Port 1 Down	All links in Port 1 are down
0	2	4	00h	Link 0 Down	Link 0 lost and cannot regain sync
0	3	4	20h	Link 1 Down	Link 1 lost and cannot regain sync
0	4	4	01h	Link 0 Fragile	Link 0 has only one or two pairs working
0	5	4	21h	Link 1 Fragile	Link 1 has only one or two pairs working
0	6	4	02h	Link 0 Up	Link 0 was down and just re-synchronized
0	7	4	22h	Link 1 Up	Link 1 was down and just re-synchronized
0	8	4	03h	Link 0 CRC Counter Wrap	–
0	9	4	04h	Link 0 8B/10B Counter Wrap	–
0	10	–	–	Reserved	–
0	11	4	06h	Link 0 Frame Count Wrap	–
0	12	4	07h	Link 0 Line Count Wrap	–
0	13	4	08h	Link 0 Empty Line Wrap	–
0	14	4	23h	Link 1 CRC Counter Wrap	–
0	15	4	24h	Link 1 8B/10B Counter Wrap	–
0	16	–	–	Reserved	–
0	17	4	26h	Link 1 Frame Count Wrap	–
0	18	4	27h	Link 1 Line Count Wrap	–
0	19	4	28h	Link 1 Empty Line Wrap	–
0	20	–	–	Reserved	–
0	21	–	–	Reserved	–
0	22	6	0	Destination Channel Address Range Error	Multicast or path-routed range check failed
0	23	6	1	Destination Channel Path Protection Error	Multicast or path-routed path/group check failed
0	24	6	2	Invalid Destination Channel ID	Dest. Channel > 7 and not 255
0	25	6	3	Address Routing Failure (non-Cfg) (from Fabric)	I/O or Memory upstream (from fabric) address-routed inverse decode failure at root
0	26	27	8	Fabric PCI special cycle	Cfg T1 to PCI Special Cycle detected in fabric
0	27	8	0	SGF Done	–
0	28	9	0	Event Overrun	Routing event cache full – cannot accept event
0	29	27	7	Address Routing Failure (no path – from PCI)	I/O or Memory downstream address-routed port map decode failure at root or upstream failure at leaf (from PCI)
0	30	–	–	Reserved	–
0	31	–	–	Reserved	–
1	0	10	0	SERR Event: Address parity error	–

Table 4–13 Event Status Bit Assignments *(Continued)*

Dword	Dword Bit	PEC	SEC	Event Bit	Set Condition
1	1	10	1	SERR Event: Master abort on write w/o ack	–
1	2	10	2	SERR Event: Target abort on write w/o ack	–
1	3	10	3	SERR Event: Parity error on write w/o ack	–
1	4	10	4	SERR Event: Target response timer expired	2^{25} cycles with no TRDY_L, master or target abort in response to PCI transaction
1	5	10	5	Response frame timer expired	2^{32} cycles with no response frame detected in response to delayed transaction
1	6	10	6	SERR Event: Delayed transaction master time-out	Master has not reattempted delayed PCI transaction
1	7	–	–	Reserved	–
1	8	11	0	PCI Status: Detected Parity Error	Detected Parity Error bit set in Status or Secondary Status register
1	9	11	1	PCI Status: Signaled System Error	Signaled System Error bit set in Status register
1	10	11	2	PCI Status: Received Master Abort	Received Master Abort bit set in Status or Secondary Status register
1	11	11	3	PCI Status: Received Target Abort	Received Target Abort bit set in Status or Secondary Status register
1	12	11	4	PCI Status: Signaled Target Abort	Signaled Target Abort bit set in Status or Secondary Status register
1	13	11	5	PCI Status: Master Data Parity Error	Master Data Parity Error bit set in Status or Secondary Status register
1	14	11	6	PCI Status: Received System Error	Received System Error set in Secondary Status register
1	15	11	7	PCI Status: P2P Master Discard Timer	Master Discard Timer Error set in Bridge Control register
1	16	27	000h	Source Channel 0 Address Range Error	Source Channel 0 address range check failed
1	17	27	080h	Source Channel 1 Address Range Error	Source Channel 1 address range check failed
1	18	27	100h	Source Channel 2 Address Range Error	Source Channel 2 address range check failed
1	19	27	180h	Source Channel 3 Address Range Error	Source Channel 3 address range check failed
1	20	27	200h	Source Channel 4 Address Range Error	Source Channel 4 address range check failed
1	21	27	280h	Source Channel 5 Address Range Error	Source Channel 5 address range check failed
1	22	27	300h	Source Channel 6 Address Range Error	Source Channel 6 address range check failed
1	23	27	380h	Source Channel 7 Address Range Error	Source Channel 7 address range check failed
1	24	27	1	Invalid segment table entry – frame discarded	Segment selected whose path length is set to 8 or higher (invalid)
1	25	27	2	Response frame: no transaction number match	Response frame request transaction number is greater than 7 and less than 61
1	26	–	–	Reserved	–
1	27	27	4	Non-existent output port 1 specified	Segment entry selects Port 1 and there is no Port 1 (bundled port or Link 1 down)
1	28	27	5	Parity Error on Provisioning Write	Parity error detected on incoming PCI write to be translated to provisioning frame (frame dropped)

Table 4–13 Event Status Bit Assignments *(Continued)*

Dword	Dword Bit	PEC	SEC	Event Bit	Set Condition
1	29	27	6	Multicast: Distribution Error	Multicast send failed due to: 1. No members in group 2. Port down 3. Port 1 selected and there is no Port 1
1	30	28	0	Segment Table Entry Invalidated	Path invalidation resulted in one or more entries set invalid
1	31	–	–	Reserved	–

4.6.8.8 Event Dispatch Control

Ch. 255 Byte Offset	5120h:5123h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
0	CETENA	R/W	0	Chip Event Table Enable. When 0 The SG2010 uses default values rather than the Chip Event Table to generate chip event frames. 1 The SG2010 uses the Chip Event Table to generate chip event frames.
1	SETENA1	R/W	0	Signal Event Table Enable for INTA_L. When: 0 The SG2010 uses default values rather than the Signal Event Table to generate event frames based on INTA_L assertion/deassertion. 1 The SG2010 uses the Signal Event Table to generate event frames when INTA_L asserts or deasserts.
2	SETENA2	R/W	0	Signal Event Table Enable for INTB_L. When: 0 The SG2010 uses default values rather than the Signal Event Table to generate event frames based on INTB_L assertion/deassertion. 1 The SG2010 uses the Signal Event Table to generate event frames when INTB_L asserts or deasserts.
3	SETENA3	R/W	0	Signal Event Table Enable for INTC_L. When: 0 The SG2010 uses default values rather than the Signal Event Table to generate event frames based on INTC_L assertion/deassertion. 1 The SG2010 uses the Signal Event Table to generate event frames when INTC_L asserts or deasserts.
4	SETENA4	R/W	0	Signal Event Table Enable for INTD_L. When: 0 The SG2010 uses default values rather than the Signal Event Table to generate event frames based on INTD_L assertion/deassertion. 1 The SG2010 uses the Signal Event Table to generate event frames when INTD_L asserts or deasserts.

Bit	Name	Access	Reset Value	Description
5	SETENA5	R/W	0	Signal Event Table Enable for PME_L. When: 0 The SG2010 uses default values rather than the Signal Event Table to generate event frames based on PME_L assertion/deassertion. 1 The SG2010 uses the Signal Event Table to generate event frames when PME_L asserts or deasserts.
6	SETENA6	R/W	0	Signal Event Table Enable for ENUM_L. When: 0 The SG2010 uses default values rather than the Signal Event Table to generate event frames based on ENUM_L assertion/deassertion. 1 The SG2010 uses the Signal Event Table to generate event frames when ENUM_L asserts or deasserts.
7	SETENA7	R/W	0	Signal Event Table Enable for SERR_L. When: 0 The SG2010 uses default values rather than the Signal Event Table to generate event frames based on SERR_L assertion/deassertion. 1 The SG2010 uses the Signal Event Table to generate event frames when SERR_L asserts or deasserts.
8	MASKIN0	R/W	0* 1†	INTA_L input signal mask. When: 0 The SG2010 detects INTA_L assertions and deassertions and generates INTA_L signal events. 1 The SG2010 ignores INTA_L as an input.
9	MASKIN1	R/W	0* 1†	INTB_L input signal mask. When: 0 The SG2010 detects INTB_L assertions and deassertions and generates INTB_L signal events. 1 The SG2010 ignores INTB_L as an input.
10	MASKIN2	R/W	0* 1†	INTC_L input signal mask. When: 0 The SG2010 detects INTC_L assertions and deassertions and generates INTC_L signal events. 1 The SG2010 ignores INTC_L as an input.
11	MASKIN3	R/W	0* 1†	INTD_L input signal mask. When: 0 The SG2010 detects INTD_L assertions and deassertions and generates INTD_L signal events. 1 The SG2010 ignores INTD_L as an input.
12	MASKIN4	R/W	0* 1†	PME_L input signal mask. When: 0 The SG2010 detects PME_L assertions and deassertions and generates PME_L signal events. 1 The SG2010 ignores PME_L as an input.
13	MASKIN5	R/W	0* 1†	ENUM_L input signal mask. When: 0 The SG2010 detects ENUM_L assertions and deassertions and generates ENUM_L signal events. 1 The SG2010 ignores ENUM_L as an input.

Bit	Name	Access	Reset Value	Description
14	MASKIN6	R/W	0[*] 1[†]	SERR_L input signal mask. When: 0 The SG2010 detects SERR_L assertions and generates SERR_L signal events 1 The SG2010 ignores SERR_L as an input.
15	RES	R	0	Reserved
16	INTASIG	R	H/W	Reflects the current active-high (inverted) signal value of INTA_L
17	INTBSIG	R	H/W	Reflects the current active-high (inverted) signal value of INTB_L
18	INTCSIG	R	H/W	Reflects the current active-high (inverted) signal value of INTC_L
19	INTDSIG	R	H/W	Reflects the current active-high (inverted) signal value of INTD_L
20	PMESIG	R	H/W	Reflects the current active-high (inverted) signal value of PME_L
21	ENMSIG	R	H/W	Reflects the current active-high (inverted) signal value of ENUM_L
31:22	RES	R	0	Reserved

* When the SG2010 is a leaf and the Bridge function is enabled.

† When the SG2010 is a root, or is a leaf in the Gateway-only mode (Bridge disabled).

4.6.8.9 Event Handler Control

Ch. 255 Byte Offset	51F0h:51F3h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
0	MSIENA0	R/W	1[*]	MSI Enable for EMU0 (local INTA_L). When: 0 Or if the Gateway MSI unit is not enabled, INTA_L is asserted for these interrupts (if enabled). 1 And if the Gateway MSI unit is enabled, MSI messages are sent instead of asserting INTA_L for this EMU.
1	MSIENA1	R/W	0	MSI Enable for EMU1 (remote INTA_L). When: 0 Or if the Gateway MSI unit is not enabled, INTA_L is asserted for these interrupts (if enabled). 1 And if the MSI unit is enabled, MSI messages are sent instead of asserting INTA_L for this EMU.
2	MSIENA2	R/W	0	MSI Enable for EMU2 (INTB_L). When: 0 Or if the Gateway MSI unit is not enabled, INTB_L is asserted for these interrupts (if enabled). 1 And if the MSI unit is enabled, MSI messages are sent instead of asserting INTB_L for this EMU.

Bit	Name	Access	Reset Value	Description
3	MSIENA3	R/W	0	MSI Enable for EMU3 (local INTC_L). When: 0 Or if the Gateway MSI unit is not enabled, INTC_L is asserted for these interrupts (if enabled). 1 And if the MSI unit is enabled, MSI messages are sent instead of asserting INTC_L for this EMU.
4	MSIENA4	R/W	0	MSI Enable for EMU4 (local INTD_L). When: 0 Or if the Gateway MSI unit is not enabled, INTD_L is asserted for these interrupts (if enabled). 1 And if the MSI unit is enabled, MSI messages are sent instead of asserting INTD_L for this EMU.
15:5	RES	R	0	Reserved
16	INTMASK0	R/W	0	Interrupt mask for local INTA_L.When: 0 Assertion of INTA_L due to EMU0 counter ≠ 0 is enabled. 1 Assertion of INTA_L due to EMU0 counter ≠ 0 is disabled.
17	INTMASK1	R/W	0	Interrupt mask for INTA_L. When: 0 Assertion of INTA_L due to EMU1 counter ≠ 0 is enabled. 1 Assertion of INTA_L due to EMU1 counter ≠ 0 is disabled.
18	INTMASK2	R/W	0	Interrupt mask for INTB_L. When: 0 Assertion of INTB_L due to EMU2 counter ≠ 0 is enabled. 1 Assertion of INTB_L due to EMU2 counter ≠ 0 is disabled.
19	INTMASK3	R/W	0	Interrupt mask for INTC_L. When: 0 Assertion of INTC_L due to EMU3 counter ≠ 0 is enabled. 1 Assertion of INTC_L due to EMU3 counter ≠ 0 is disabled.
20	INTMASK4	R/W	0	Interrupt mask for INTD_L. When: 0 Assertion of INTD_L due to EMU4 counter ≠ 0 is enabled. 1 Assertion of INTD_L due to EMU4 counter ≠ 0 is disabled.
21	INTMASK5	R/W	0	Interrupt mask for PME_L. When: 0 Assertion of PME_L due to EMU5 counter ≠ 0 is enabled. 1 Assertion of PME_L due to EMU5 counter ≠ 0 is disabled.

Bit	Name	Access	Reset Value	Description
22	INTMASK6	R/W	0	Interrupt mask for ENUM_L. When: 0 Assertion of ENUM_L due to EMU6 counter ≠ 0 is enabled. 1 Assertion of ENUM_L due to EMU6 counter ≠ 0 is disabled.
23	INTMASK7	R/W	0	Interrupt mask for SERR_L. When: 0 Assertion of SERR_L due to EMU7 counter ≠ 0 is enabled. 1 Assertion of SERR_L due to EMU7 counter ≠ 0 is disabled.
24	INTMASK8	R/W	0	Reserved Interrupt mask. Not associated with any signal in the SG2010.
31:25	RES	R	0	Reserved

* This is a reset value of 1. If the MSI unit is enabled, by default the SG2010 uses MSI only for local events. For events initiated by other devices, the MSI units of those devices should be used. However, if desired, software can use bits 4:1 to send MSI messages for remote signal events.

4.6.8.10 Path Invalidation Control

Ch. 255 Byte Offset	51F4h: 51F7h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
0	PIDIS	R/W1TC	0	Disable Last Path Invalidation. When 1, the SG2010 does not perform path invalidation when a path event is received that has a path matching the path returned in this register. The SG2010 still performs invalidation on other paths. When 0, the SG2010 performs path invalidation when a path event is received with the path returned in this register (or any other path). The SG2010 hardware sets this bit when it performs a path invalidation and writes the path invalidated into the path field. Software clears this bit when it wishes to enable the SG2010 to perform path invalidation when this path is received in a path event frame.
4:1	RES	R	0	Reserved
5	PIINPORT	R	0	Input port of the received path event frame.
7:6	RES	R	0	Reserved
31:8	PIPATH	R	0	Last Path Invalidated. Contains the last path invalidated. Bits [31:11] contain the transformed path turns 0 ([13:11]) through 7 ([31:29]) and bits [10:8] contain the turn count.

4.6.8.11 Event Message Buffer Upper Base Address

Ch. 255 Byte Offset	51F8h:51FBh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	BUBASE	R/W	0	Upper base address for all event message buffers. Specifies PCI address bits [63:32] for event messages generated by all Event Message Units.

4.6.8.12 Event Message Buffer Size

Ch. 255 Byte Offset	51FCh:51FFh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
1:0	BSIZE0	R/W	0	Sets the size of the EMU0 event message buffer in memory. Controls the number of base address and tail pointer bits for the event message PCI address. Possible values are: 00b: Disabled 01b: 4KB 10b: 16KB 11b: 64KB
3:2	BSIZE1	R/W	0	Sets the size of the EMU1 event message buffer in memory. Controls the number of base address and tail pointer bits for the event message PCI address. Possible values are: 00b: Disabled 01b: 4KB 10b: 16KB 11b: 64KB
5:4	BSIZE2	R/W	0	Sets the size of the EMU2 event message buffer in memory. Controls the number of base address and tail pointer bits for the event message PCI address. Possible values are: 00b: Disabled 01b: 4KB 10b: 16KB 11b: 64KB
7:6	BSIZE3	R/W	0	Sets the size of the EMU3 event message buffer in memory. Controls the number of base address and tail pointer bits for the event message PCI address. Possible values are: 00b: Disabled 01b: 4KB 10b: 16KB 11b: 64KB

Bit	Name	Access	Reset Value	Description
9:8	BSIZE4	R/W	0	Sets the size of the EMU4 event message buffer in memory. Controls the number of base address and tail pointer bits for the event message PCI address. Possible values are: 00b: Disabled 01b: 4KB 10b: 16KB 11b: 64KB
11:10	BSIZE5	R/W	0	Sets the size of the EMU5 event message buffer in memory. Controls the number of base address and tail pointer bits for the event message PCI address. Possible values are: 00b: Disabled 01b: 4KB 10b: 16KB 11b: 64KB
13:12	BSIZE6	R/W	0	Sets the size of the EMU6 event message buffer in memory. Controls the number of base address and tail pointer bits for the event message PCI address. Possible values are: 00b: Disabled 01b: 4KB 10b: 16KB 11b: 64KB
15:14	BSIZE7	R/W	0	Sets the size of the EMU7 event message buffer in memory. Controls the number of base address and tail pointer bits for the event message PCI address. Possible values are: 00b: Disabled 01b: 4KB 10b: 16KB 11b: 64KB
17:16	BSIZE8	R/W	0	Sets the size of the EMU8 event message buffer in memory. Controls the number of base address and tail pointer bits for the event message PCI address. Possible values are: 00b: Disabled 01b: 4KB 10b: 16KB 11b: 64KB
31:18	RES	R	0	Reserved

4.6.8.13 Event Message Unit Registers

4.6.8.13.1 EMU*x* Counter Increment/Event Message Write

EMU0 Ctr Incr/Event Msg Write Ch. 255 Byte Offset	5200h:5203h
EMU1 Ctr Incr/Event Msg Write Ch. 255 Byte Offset	5208h:520Bh
EMU2 Ctr Incr/Event Msg Write Ch. 255 Byte Offset	5210h:5213h

EMU3 Ctr Incr/Event Msg Write Ch. 255 Byte Offset	5218h:521Bh
EMU4 Ctr Incr/Event Msg Write Ch. 255 Byte Offset	5220h:5223h
EMU5 Ctr Incr/Event Msg Write Ch. 255 Byte Offset	5228h:522Bh
EMU6 Ctr Incr/Event Msg Write Ch. 255 Byte Offset	5230h:5233h
EMU7 Ctr Incr/Event Msg Write Ch. 255 Byte Offset	5238h:523Bh
EMU8 Ctr Incr/Event Msg Write Ch. 255 Byte Offset	5240h:5243h
Size	4 bytes per register

Bit	Name	Access	Reset Value	Description
31:0	EMUADRx	R/W	0	When this register is written, the event counter associated with this EMU is incremented and, if enabled, a single Dword write message (using the write data) is written to the associated event message buffer. The interrupt signal controlled by the event counter is asserted if it is unmasked. When this register is read, this location returns the value of the event counter for this EMU. Each counter is 13 bits wide. Signal associations are described in Section 3.7.3 This event counter is also incremented, and an event message written to PCI, by incoming event frames targeting the corresponding EMU address.

4.6.8.13.2 EMU*x* Counter Decrement

EMU0 Counter Decrement Ch. 255 Byte Offset	5204h:5207h
EMU1 Counter Decrement Ch. 255 Byte Offset	520Ch:520Fh
EMU2 Counter Decrement Ch. 255 Byte Offset	5214h:5217h
EMU3 Counter Decrement Ch. 255 Byte Offset	521Ch:521Fh
EMU4 Counter Decrement Ch. 255 Byte Offset	5224h:5227h
EMU5 Counter Decrement Ch. 255 Byte Offset	522Ch:522Fh
EMU6 Counter Decrement Ch. 255 Byte Offset	5234h:5237h
EMU7 Counter Decrement Ch. 255 Byte Offset	523Ch:523Fh
EMU8 Counter Decrement Ch. 255 Byte Offset	5244h:5247h
Size	4 bytes per register

Bit	Name	Access	Reset Value	Description
31:0	EMUADRx	R/W	0	When this register is written, the event counter associated with this EMU is decremented. If the counter decrements to 0, the interrupt signal controlled by the event counter is deasserted. When this register is read, this location returns the value of the event counter. Each counter is 13 bits wide. Signal associations are described in Section 3.7.3 This event counter is also decremented by incoming event frames targeting the corresponding EMU address.

4.6.8.14 EMUx Event Message Buffer Tail Pointer

EMU0 Tail Pointer Ch. 255 Byte Offset	5280h:5283h
EMU1 Tail Pointer Ch. 255 Byte Offset	5284h:5287h
EMU2 Tail Pointer Ch. 255 Byte Offset	5288h:528Bh
EMU3 Tail Pointer Ch. 255 Byte Offset	528Ch:528Fh

EMU4 Tail Pointer Ch. 255 Byte Offset	5290h:5293h
EMU5 Tail Pointer Ch. 255 Byte Offset	5294h:5297h
EMU6 Tail Pointer Ch. 255 Byte Offset	5298h:529Bh
EMU7 Tail Pointer Ch. 255 Byte Offset	529Ch:529Fh
EMU8 Tail Pointer Ch. 255 Byte Offset	52A0h:52A3h
Size	4 bytes per register, 9 registers

Bit	Name	Access	Reset Value	Description
2:0	RES	R	0	Indicates the three LSBs of the event message buffer address for the corresponding EMU. These bits always read as 0 because event message buffer addresses are always quadword aligned.
16:3	EMUTP	R/W	0	Specifies the quadword-aligned tail pointer for the corresponding EMU event message buffer. The tail pointer is combined with the corresponding base address to produce a PCI address for the event message. After an event message is written, the tail pointer increments by 8 bytes. The number of bits used for the tail pointer depends on the size of the event message buffer; however the counter always wraps at 14 bits.
31:17	RES	R	0	Reserved

4.6.8.15 EMU x Event Message Lower Base Address

EMU0 Lower Base Address Ch. 255 Byte Offset	52C0h:52C3h
EMU1 Lower Base Address Ch. 255 Byte Offset	52C4h:52C7h
EMU2 Lower Base Address Ch. 255 Byte Offset	52C8h:52CBh
EMU3 Lower Base Address Ch. 255 Byte Offset	52CCh:52CFh
EMU4 Lower Base Address Ch. 255 Byte Offset	52D0h:52D3h
EMU5 Lower Base Address Ch. 255 Byte Offset	52D4h:52D7h
EMU6 Lower Base Address Ch. 255 Byte Offset	52D8h:52DBh
EMU7 Lower Base Address Ch. 255 Byte Offset	52DCh:52DFh
EMU8 Lower Base Address Ch. 255 Byte Offset	52E0h:52E3h
Size	4 bytes per register, 9 registers

Bit	Name	Access	Reset Value	Description
11:0	RES	R	0	Reserved
31:12	BLBASE	R/W	0	Specifies the lower base address of the event message buffer address for the corresponding EMU. The bits used depend on the size of the event message buffer, as follows: 4KB: bits [31:12] 16KB: bits [31:14] 64KB: bits [31:16] The remaining bits are ignored by the SG2010.

4.6.9 Software Generated Frame Registers

These register provide the ability for software to generate StarFabric frames that are sent into the fabric.

4.6.9.1 Frame

The Frame registers hold up to 16 bytes of header information and up to128 data bytes for a maximum 144-byte, or nine-line frame. The header is included in the first line, followed by additional data payload, if any. The SG2010 attaches the link overhead to the frame according to the Link Overhead bit written as a part of the frame header.

Line 0 Dword 1 Ch. 255 Byte Offset	5600h:5603h
Line 0 Dword 2 Ch. 255 Byte Offset	5604h:5607h
Line 0 Dword 3 Ch. 255 Byte Offset	5608h:560Bh
Line 0 Dword 4 Ch. 255 Byte Offset	560Ch:560Fh
Line 1 Dword 1 Ch. 255 Byte Offset	5610h:5613h
Line 1 Dword 2 Ch. 255 Byte Offset	5614h:5617h
Line 1 Dword 3 Ch. 255 Byte Offset	5618h:561Bh
Line 1 Dword 4 Ch. 255 Byte Offset	561Ch:561Fh
Line 2 Dword 1 Ch. 255 Byte Offset	5620h:5623h
Line 2 Dword 2 Ch. 255 Byte Offset	5624h:5627h
Line 2 Dword 3 Ch. 255 Byte Offset	5628h:562Bh
Line 2 Dword 4 Ch. 255 Byte Offset	562Ch:562Fh
Line 3 Dword 1 Ch. 255 Byte Offset	5630h:5633h
Line 3 Dword 2 Ch. 255 Byte Offset	5634h:5637h
Line 3 Dword 3 Ch. 255 Byte Offset	5638h:563Bh
Line 3 Dword 4 Ch. 255 Byte Offset	563Ch:563Fh
Line 4 Dword 1 Ch. 255 Byte Offset	5640h:5643h
Line 4 Dword 2 Ch. 255 Byte Offset	5644h:5647h
Line 4 Dword 3 Ch. 255 Byte Offset	5648h:564Bh
Line 4 Dword 4 Ch. 255 Byte Offset	564Ch:564Fh
Line 5 Dword 1 Ch. 255 Byte Offset	5650h:5653h
Line 5 Dword 2 Ch. 255 Byte Offset	5654h:5657h
Line 5 Dword 3 Ch. 255 Byte Offset	5658h:565Bh
Line 5 Dword 4 Ch. 255 Byte Offset	565Ch:565Fh
Line 6 Dword 1 Ch. 255 Byte Offset	5660h:5663h
Line 6 Dword 2 Ch. 255 Byte Offset	5664h:5667h
Line 6 Dword 3 Ch. 255 Byte Offset	5668h:566Bh
Line 6 Dword 4 Ch. 255 Byte Offset	566Ch:566Fh
Line 7 Dword 1 Ch. 255 Byte Offset	5670h:5673h
Line 7 Dword 2 Ch. 255 Byte Offset	5674h:5677h
Line 7 Dword 3 Ch. 255 Byte Offset	5678h:567Bh
Line 7 Dword 4 Ch. 255 Byte Offset	567Ch:567Fh
Line 8 Dword 1 Ch. 255 Byte Offset	5680h:5683h
Line 8 Dword 2 Ch. 255 Byte Offset	5684h:5687h
Line 8 Dword 3 Ch. 255 Byte Offset	5688h:568Bh
Line 8 Dword 4 Ch. 255 Byte Offset	568Ch:568Fh
Size	144 bytes

Bit	Name	Access	Reset Value	Description
31:0	FD*x*L*x*	R/W	0	Contains a Dword of frame data to be sent as a software generated frame.

4.6.9.2 SGF Destination Address

The SGF Destination Address specifies the Dword-aligned address in local memory to which the SG2010 writes response frames that it receives in response to SGFs. When a response frame is written, the SG2010 adjusts the address pointer to the next 16-byte (line) aligned address boundary. The address pointer wraps at a 16KB boundary. A 16KB buffer can hold up to 1K single line frames.

Ch. 255 Byte Offset	5698h:569Fh
Size	8 bytes

Bit	Name	Access	Reset Value	Description
1:0	RES	R	0	Reserved.
3:2	SGF_DADR	R	0	Destination address bits [3:2]. These bits give the Dword index within a frame line. These bits are reset to 0 when a response frame has been written to PCI. These bits increment as each read completion Dword is written.
13:4	SGF_LADR	R/W		Destination address bits [13:4]. These bits give the line address within the memory buffer used to store read completions. This field is incremented every time a frame line is written to memory; in other words, when the fourth Dword in a line is written. This field wraps to 0 on overflow (starts writing at the beginning of the 16K buffer).
63:14	SGF_BADR	R/W	0	Specifies the PCI base address for a 16K memory buffer to store read completion frames. The SG2010 does not modify this field.

4.6.9.3 SGF Control and Status Register

Ch. 255 Byte Offset	56A0h:56A3h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
0	SEND_SGF	R/W1TS	0	When set to 1, the SG2010 sends the data in the Frame registers out of the appropriate link. The SG2010 clears this bit after the frame has been sent or discarded. Writing 0 to this bit has no effect.
1	RESP_OUT	R/W1TC	0	Outstanding response. This bit is set by the SG2010 when an SGF frame requiring a response frame is sent, and cleared by the SG2010 when it receives a write acknowledge, bandwidth response, or the last read completion frame. When software writes 1 to this bit, it is cleared and the SG2010 SGF function is reset; that is, it no longer waits for a response frame.

Bit	Name	Access	Reset Value	Description
2	SGF_DONE	R	0	The SG2010 sets this bit to 1 when it receives the last response frame in response to an SGF and has stored it in local memory. If a response frame is not required, then the SG2010 sets the SGF_DONE bit when the frame is sent. This bit is also set if the SGF is discarded and not sent due to an error. The SG2010 clears this bit when software writes 1 to the SGF_DONE bit in the Raw Event Status register or when the SEND_SGF bit is set to send another SGF.
3	SGF_NSNT	R	0	The SG2010 sets this bit to 1 when 1 is written to the SEND_SGF bit, but the selected output link is down or an error is encountered. The SG2010 clears this bit when the SEND_SGF bit is set and the frame is successfully sent.
7:4	CMPSTAT	R	0	Response frame failure status. When the SG2010 receives a read completion or a write acknowledge, the Failure Type from the header is copied to this register field. When the SG2010 receives a bandwidth response frame, the Secondary Operation field (containing the operation status) is copied to this register field. This status remains available until the next read completion or write acknowledge is received.
8	RES	R	0	Reserved.
9	SGFOUT	R/W	0	SGF Output Link. Selects the exit link 0 or 1 for the SGF.
15:10	RES	R	0	Reserved.
16	WADIS	R/W	0	Disables delivering SGF write acknowledge frames and bandwidth response frames to memory. When: 0 Write acknowledge frames and bandwidth response frames are written to memory. 1 SGF write acknowledge frames and bandwidth response frames are not written to memory. The completion status is still available in this register.
31:17	RES	R	0	Reserved.

4.6.9.4 SGF Bytes Received

Ch. 255 Byte Offset	56A4h:56A7h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
1:0	RES	R	0	The two LSBs of the bytes received field. These two bits always read as 0 because Dword granularity is used when writing out responses to memory.
12:2	SGFBR	R/W	0	Returns the byte count [12:1] of the last read completion response to an SGF that has been stored in local memory. Writable for diagnostic purposes. The SG2010 resets this counter to zero when the SEND_SGF bit is written.
31:13	RES	R	0	Reserved

4.6.10 Source Channel Table

The Source Channel Table holds the channel properties for eight source channels. There are eight copies of each source channel register, one for each channel number (0 – 7) supported by the SG2010.

4.6.10.1 Source Channel *N* Translation Address

Source Channel 0 Ch. 255 Byte Offset	5C00h:5C07h
Source Channel 1 Ch. 255 Byte Offset	5C10h:5C17h
Source Channel 2 Ch. 255 Byte Offset	5C20h:5C27h
Source Channel 3 Ch. 255 Byte Offset	5C30h:5C37h
Source Channel 4 Ch. 255 Byte Offset	5C40h:5C47h
Source Channel 5 Ch. 255 Byte Offset	5C50h:5C57h
Source Channel 6 Ch. 255 Byte Offset	5C60h:5C67h
Source Channel 7 Ch. 255 Byte Offset	5C70h:5C77h
Size	8 bytes

Bit	Name	Access	Reset Value	Description
2:0	RES	R/W	X	Reserved. These bits do not control any functionality, but are R/W.
11:3	RES	R	0	Reserved
43:12	SC*n*TWA	R/W	X	Specifies bits [43:12] of a base address for a source channel address translation when converting a PCI transaction to a path-routed or multicast frame. The base address is 4KB-aligned, and is added to the outgoing address before it goes out the link.
63:44	RES	R	0	Reserved.

4.6.10.2 Source Channel *N* Address Range

Source Channel 0 Ch. 255 Byte Offset	5C08h:5C0Dh
Source Channel 1 Ch. 255 Byte Offset	5C18h:5C1Dh
Source Channel 2 Ch. 255 Byte Offset	5C28h:5C2Dh
Source Channel 3 Ch. 255 Byte Offset	5C38h:5C3Dh
Source Channel 4 Ch. 255 Byte Offset	5C48h:5C4Dh

Source Channel 5 Ch. 255 Byte Offset	5C58h:5C5Dh
Source Channel 6 Ch. 255 Byte Offset	5C68h:5C6Dh
Source Channel 7 Ch. 255 Byte Offset	5C78h:5C7Dh
Size	6 bytes

Bit	Name	Access	Reset Value	Description
11:0	RES	R	0	Reserved
43:12	SC*n*AR	R/W	X	Specifies bits [43:12] for a source channel address range check when converting a PCI transaction to a path-routed or multicast frame. If the source address before address translation exceeds this value, the SG2010 signals a Source Channel Address Range Error event.
47:44	RES	R	0	Reserved.

4.6.10.3 Source Channel *N* Control

Source Channel 0 Ch. 255 Byte Offset	5C0Eh:5C0Fh
Source Channel 1 Ch. 255 Byte Offset	5C1Eh:5C1Fh
Source Channel 2 Ch. 255 Byte Offset	5C2Eh:5C2Fh
Source Channel 3 Ch. 255 Byte Offset	5C3Eh:5C3Fh
Source Channel 4 Ch. 255 Byte Offset	5C4Eh:5C4Fh
Source Channel 5 Ch. 255 Byte Offset	5C5Eh:5C5Fh
Source Channel 6 Ch. 255 Byte Offset	5C6Eh:5C6Fh
Source Channel 7 Ch. 255 Byte Offset	5C7Eh:5C7Fh
Size	2 bytes

Bit	Name	Access	Reset Value	Description
9:0	SC*n*PRA	R/W	X	Specifies the number of Dwords to be read from the target for a prescriptive read. The SG2010 supports prescriptive read requests from 1 to 512 Dwords. Other values should not be programmed and results are undefined. Possible values are: 000h: Illegal 001h: 1 Dwords 002h: 2 Dwords 003h: 3 Dwords … … 200h: 512 Dwords 201h – 3FFh: Illegal
10	PRE_ENA	R/W	X	Prescriptive Read Enable. When: 0 A speculative read command is used and the amount of Dwords requested is based on command type and cache line size. 1 A prescriptive read command is used for all reads using this source channel.
15	WA_ENA	R/W	0	Write Acknowledge Enable. When: 0 A write without acknowledge operation is performed. 1 A write with acknowledge operation is used for path-routed or multicast write frames using this source channel.

4.6.11 Destination Channel Table

The Destination Channel Table holds the properties for eight destination channels. There are eight copies of each destination channel register, one for each destination channel (0 – 7) supported by the SG2010.

4.6.11.1 Destination Channel *N* Translation Address

Destination Channel 0 Ch. 255 Byte Offset	5D00h:5D07h
Destination Channel 1 Ch. 255 Byte Offset	5D20h:5D27h
Destination Channel 2 Ch. 255 Byte Offset	5D40h:5D47h
Destination Channel 3 Ch. 255 Byte Offset	5D60h:5D67h
Destination Channel 4 Ch. 255 Byte Offset	5D80h:5D87h
Destination Channel 5 Ch. 255 Byte Offset	5DA0h:5DA7h
Destination Channel 6 Ch. 255 Byte Offset	5DC0h:5DC7h
Destination Channel 7 Ch. 255 Byte Offset	5DE0h:5DE7h
Size	8 bytes

Bit	Name	Access	Reset Value	Description
11:0	RES	R	0	Reserved
63:12	DC*n*TA	R/W	0	Specifies base address bits [63:12] for a destination channel address translation when converting a path-routed or multicast frame to a PCI transaction. The base address is 4KB-aligned, and is added to the incoming frame offset to create a PCI address. An overflow on the translation causes the resulting address to wrap.

4.6.11.2 Destination Channel *N* Offset Range

Destination Channel 0 Ch. 255 Byte Offset	5D08h:5D0Dh
Destination Channel 1 Ch. 255 Byte Offset	5D28h:5D2Dh
Destination Channel 2 Ch. 255 Byte Offset	5D48h:5D4Dh
Destination Channel 3 Ch. 255 Byte Offset	5D68h:5D6Dh
Destination Channel 4 Ch. 255 Byte Offset	5D88h:5D8Dh
Destination Channel 5 Ch. 255 Byte Offset	5DA8h:5DADh
Destination Channel 6 Ch. 255 Byte Offset	5DC8h:5DCDh
Destination Channel 7 Ch. 255 Byte Offset	5DE8h:5DEDh
Size	6 bytes

Bit	Name	Access	Reset Value	Description
11:0	RES	R	0	Reserved
43:12	DC*n*AC	R/W	0	Specifies bits [43:12] for a destination channel offset range check when converting a path-routed or multicast frame to a PCI transaction. If the incoming frame offset exceeds this value before translation, the SG2010 signals a Destination Channel Range event.
47:44	RES	R	0	Reserved.

4.6.11.3 Destination Channel *N* Control

Destination Channel 0 Ch. 255 Byte Offset	5D0Eh:5D0Fh
Destination Channel 1 Ch. 255 Byte Offset	5D2Eh:5D2Fh
Destination Channel 2 Ch. 255 Byte Offset	5D4Eh:5D4Fh
Destination Channel 3 Ch. 255 Byte Offset	5D6Eh:5D6Fh
Destination Channel 4 Ch. 255 Byte Offset	5D8Eh:5D8Fh
Destination Channel 5 Ch. 255 Byte Offset	5DAEh:5DAFh
Destination Channel 6 Ch. 255 Byte Offset	5DCEh:5DCFh
Destination Channel 7 Ch. 255 Byte Offset	5DEEh:5DEFh
Size	2 bytes

Bit	Name	Access	Reset Value	Description
0	DC*n*PPE	R/W	0	When 0 The SG2010 does not perform any path protection checks. 1 The SG2010 performs a path protection check for either all incoming path-routed frames or all incoming multicast frames addressing this channel, depending on the type of path protection selected.
15:1	RES	R	0	Reserved.

4.6.11.4 Destination Channel *N* Path Protection 0

Destination Channel 0 Ch. 255 Byte Offset	5D10h:5D13h
Destination Channel 1 Ch. 255 Byte Offset	5D30h:5D33h
Destination Channel 2 Ch. 255 Byte Offset	5D50h:5D53h
Destination Channel 3 Ch. 255 Byte Offset	5D70h:5D73h
Destination Channel 4 Ch. 255 Byte Offset	5D90h:5D93h
Destination Channel 5 Ch. 255 Byte Offset	5DB0h:5DB3h
Destination Channel 6 Ch. 255 Byte Offset	5DD0h:5DD3h
Destination Channel 7 Ch. 255 Byte Offset	5DF0h:5DF3h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
23:0	DC*n*FPR	R/W	0	Specifies one of two allowed paths when the SG2010 is enabled to perform a path protection check. If path-routing checks are selected, all incoming path-routed frames addressing this channel must use this path or the second path (if enabled). The path consists of a turn count at bits [23:21], followed by seven three-bit turns. If multicast checks are selected, all incoming multicast frames addressing this channel must use this multicast ID or the second multicast ID (if enabled).
24	INPUT	R/W	0	This field contains the input port that accompanies the path specification. The input port used by the frame must match the input port specified in this field.

27:25	Reserved	R	0	–
28	PTHSEL	R/W	0	Selects whether this protection check is a path-routed check or a multicast check. When: 0 Bits [23:0] contain a path specification and are compared against all path-routed frames received using this channel. 1 Bits [23:0] contain a Multicast Group ID and are compared against all multicast frames received using this channel.
31:29	RES	R	0	Reserved.

4.6.11.5 Destination Channel *N* Path Protection 1

Destination Channel 0 Ch. 255 Byte Offset	5D14h:5D17h
Destination Channel 1 Ch. 255 Byte Offset	5D34h:5D37h
Destination Channel 2 Ch. 255 Byte Offset	5D54h:5D57h
Destination Channel 3 Ch. 255 Byte Offset	5D74h:5D77h
Destination Channel 4 Ch. 255 Byte Offset	5D94h:5D97h
Destination Channel 5 Ch. 255 Byte Offset	5DB4h:5DB7h
Destination Channel 6 Ch. 255 Byte Offset	5DD4h:5DD7h
Destination Channel 7 Ch. 255 Byte Offset	5DF4h:5DF7h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
23:0	DC*n*FPR	R/W	0	Specifies one of two allowed paths when the SG2010 is enabled to perform a path protection check. If path-routing checks are selected, all incoming path-routed frames addressing this channel must use this path or the first path (if enabled). The path consists of a turn count at bits [23:21], followed by seven three-bit turns. If multicast checks are selected, all incoming multicast frames addressing this channel must use this multicast ID or the first multicast ID (if enabled).
24	INPUT	R/W	0	This field contains the input port that accompanies the path specification. The input port used by the frame must match the input port specified in this field.
27:25	Reserved	R	0	–
28	PTHSEL	R/W	0	Selects whether this protection check is a path-routed check or a multicast check. When: 0 Bits [23:0] contain a path specification and are compared against all path-routed frames received using this channel. 1 Bits [23:0] contain a Multicast Group ID and are compared against all multicast frames received using this channel.
31:24	RES	R	0	Reserved.

4.6.12 Path Table

The path table consists of 128 four-byte entries. Each entry contains a path. The entire path table occupies Ch. 255 Byte Offsets 1E00h through 2000h.

Path Table Entry 0 Ch. 255 Byte Offset	5E00h:5E03h
Path Table Entry *N* Ch. 255 Byte Offset	5xxxh:5yyyh
xxx = E00 + (*N* × 4)	yyy = E03 + (*N* × 4)
Size	4 bytes per entry

Bit	Name	Access	Reset Value	Description
14:0	PATH*n*	R/W	X	Specifies Turn 1 through Turn 5 of one of 128 possible paths used by the SG2010 when converting a PCI transaction into a path-routed frame. Turn 0 and the last turn are stored in the Segment Table. The arrangement is: [2:0] – Turn 1 [5:3] – Turn 2 [8:6] – Turn 3 [11:9] – Turn 4 [14:12] – Turn 5
31:15	RES	R	0	Reserved.

4.6.13 Segment Table

The Segment Table consists of 1024 eight-byte entries. Each entry contains information needed to translate a PCI transaction into a path-routed or multicast frame. Each Segment Table entry corresponds to a PCI memory address window. The Segment Table occupies Channel 255 byte offsets 6000h through 7FFFh.

4.6.13.1 Segment Table Entry *N* Path Index/Multicast ID

Path Index 0 Ch. 255 Byte Offset	6000h
Path Table Entry *N* Ch. 255 Byte Offset	6xxxh
xxx = 8*N*	
Size	1 byte

Bit	Name	Access	Reset Value	Description
6:0	PIND*n*	R/W	X	For a path-routed frame, contains an index into the Path Table, specifying the entry to be used to obtain turn 1 through turn 6 of the path. For a multicast frame, bits [4:0] of this field specify the Multicast Group ID (0 through 31) to be used in the frame.
7	RES	R	0	Reserved.

4.6.13.2 Segment Table Entry *N* Exit Port/Path Length

Path Index 0 Ch. 255 Byte Offset	6001h
Path Table Entry *N* Ch. 255 Byte Offset	6xxxh
xxx = 1 + ($N \times 8$)	
Size	1 byte

Bit	Name	Access	Reset Value	Description
2:0	PTHL*n*	R/W	X	Specifies the path length for this segment table entry. Valid path lengths are 0h through 7h. If the CoS is Special and the Channel ID is 255, this field indicates the initial turn count of a path-routed, Channel 255, Provisioning frame. The path is shifted by the number of turns indicated by the initial turn count.
3	PVAL*n*	R/W	X	Invalid Entry. When a 1, indicates that this entry in the Segment Table is invalid; PCI transactions targeting this entry result in an error. When a 0, indicates that this is a valid Segment Table entry.
4	EXIT*n*	R/W	X	Specifies the output port of the frame, port 0 or port 1.
7:5	RES	R	0	Reserved.

4.6.13.3 Segment Table Entry *N* First and Last Turn

Path Index 0 Ch. 255 Byte Offset	6002h
Path Table Entry *N* Ch. 255 Byte Offset	6xxxh
xxx = 2 + ($N \times 8$)	
Size	1 byte

Bit	Name	Access	Reset Value	Description
2:0	TURN0*n*	R/W	X	Specifies Turn 0 for the path.
5:3	LAST*n*	R/W	X	Specifies the last *active* turn (not necessarily turn 7) for the frame's path. The last active turn is indicated by the Path Length - 1, where a Path Length of 7 indicates that Turn6 is the last active turn.
7:6	RES	R	0	Reserved

4.6.13.4 Segment Table Entry *N* Channel Control/COS

Path Index 0 Ch. 255 Byte Offset	6003h
Path Table Entry *N* Ch. 255 Byte Offset	6xxxh
xxx = 3 + (*N* × 8)	
Size	1 byte

Bit	Name	Access	Reset Value	Description
0	SRC_EN*n*	R/W	X	Source Channel Enable. When 1, indicates the source channel is associated with this segment table entry.
3:1	COS	R/W	X	Specifies the class-of-service to be used for the frame. The supported CoS values are asynchronous, isochronous, multicast, provisioning, HP-asynchronous, and HP-isochronous. Special CoS encoding is used in conjunction with Channel 255 to specify a provisioning frame with a turn count >0. All other values result in unpredictable behavior.
7:4	RES	R	0	Reserved

4.6.13.5 Segment Table Entry *N* Source Channel ID

Path Index 0 Ch. 255 Byte Offset	6004h
Path Table Entry *N* Ch. 255 Byte Offset	6xxxh
xxx = 4 + (*N* × 8)	
Size	1 byte

Bit	Name	Access	Reset Value	Description
2:0	SRCID*n*	R/W	X	Specifies one of eight (0-7) source channels to be used when the Source Channel Enable bit is set.
7:3	RES	R	0	Reserved

4.6.13.6 Segment Table Entry *N* Destination Channel ID Field

Path Index 0 Ch. 255 Byte Offset	6005h
Path Table Entry *N* Ch. 255 Byte Offset	6xxxh
xxx = 5 + (*N* × 8)	
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	DCHID*n*	R/W	X	This field specifies the Channel ID to be used in the frame.

4.6.13.7 Segment Table Entry *N* MSB Address

Path Index 0 Ch. 255 Byte Offset	6006h:6007h
Path Table Entry *N* Ch. 255 Byte Offset	6xxxh:6yyyh
xxx = 6 + (*N* × 8)	
yyy = 7 + (*N* × 8)	
Size	2 bytes

Bit	Name	Access	Reset Value	Description
8:0	MSBADDR	R/W	X	These address bits replace the PCI address bits that were used to index the Segment Table. The number of bits used is determined by the number of segments. The number of segments used is dependent on the enabled BARs and their configuration, and whether redundant routing is used.
15:9	RES	R	0	Reserved

4.7 Bridge Function Configuration Registers

4.7.1 PCI Header Registers

These registers make up the first 16 bytes of the configuration space of every PCI device.

4.7.1.1 Vendor ID

Bridge Cfg Byte Offset	00h:01h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
15:0	VENDORID	R	9902h	Identifies the vendor of this device as StarGen. Returns 9902h when read.

4.7.1.2 Device ID

Bridge Cfg Byte Offset	02h:03h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
15:0	DEVID	R	01h	PCI device identification number for the Bridge. Returns 01h when read.

4.7.1.3 Command

Bridge Cfg Byte Offset	04h:05h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
0	PIO	R/W	0	I/O Enable for primary PCI bus for the Bridge function. If the SG2010 is a root, when this bit is: 0 The SG2010 does not respond to I/O transactions on the PCI bus. 1 The SG2010 is enabled to respond to I/O transactions on the PCI bus. If the SG2010 is a leaf, a copy of this bit in SG2010's link partner's Port Map Table is used to enable downstream I/O frames.
1	PMEM	R/W	0	Memory Enable for primary PCI bus for the Bridge function. If the SG2010 is a root, when this bit is: 0 The SG2010 does not respond to memory transactions on the PCI bus. 1 The SG2010 is enabled to respond to memory transactions on the PCI bus. If the SG2010 is a leaf, a copy of this bit in SG2010's link partner's Port Map Table is used to enable downstream address-routed memory frames.
2	PBM	R/W	0	Bus Master Enable for primary PCI bus for the Bridge function. This bit controls target response to upstream I/O and Memory transactions or frames. When: 0 The SG2010 does not respond to I/O or memory transactions on the PCI bus if a leaf, or is an address routing failure if a root. 1 If the SG2010 is a leaf it is enabled to decode upstream Memory and I/O transactions or frames.
3	PSC	R	0	Special Cycles. Reads as 0; the SG2010 does not monitor special cycles on the PCI bus.
4	PMWI	R/W	0	MWI Enable for the Bridge function. When: 0 The SG2010 does not initiate Memory Write and Invalidate transactions on the PCI bus. 1 The SG2010 is enabled to initiate Memory Write and Invalidate transactions.
5	VGASN	R/W	0	VGA Snoop. When written with 1, the SG2010 forwards downstream transactions addressing the VGA I/O locations 3C6h, 3C8h, and 3C9h. Bits [15:10] are not decoded if the VGA 16-bit Decode Enable bit is 0, otherwise they are decoded as 0.

Bit	Name	Access	Reset Value	Description
6	PPER	R/W	0	Parity Error Response for the primary PCI bus for the Bridge function. When: 0 The SG2010 detects, but does not report. parity errors on the primary PCI bus (PERR_L assertion, SERR_L assertion, setting the Master Data Parity Error bit). 1 The SG2010 is enabled to report parity errors detected on the primary PCI bus.
7	STEP	R	0	Stepping Control. Reads as 0; the SG2010 does not perform stepping.
8	SERREN	R/W	0	SERR# Enable for the Bridge function. When: 0 And the Gateway SERREN is also 0, the SG2010 does not assert SERR_L. 1 The SG2010 is enabled to assert SERR_L on the primary PCI bus, or to forward SERR# in default mode when a leaf with Bridge enabled.
9	FBBENP	R/W	0	Fast Back-to-back Enable for the primary bus for the Bridge function. When: 0 The SG2010 does not perform fast back-to-back transactions on the primary PCI bus. 1 The SG2010 is enabled to generate fast back-to-back transactions on the primary PCI bus. This bit in meaningful only when the SG2010 is a root.
15:10	RES	R	0	Reserved

4.7.1.4 Status

Bridge Cfg Byte Offset	06h:07h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
3:0	RES	R	0	Reserved
4	BCAP	R	1	Capabilities List. Reads as 1 to indicate that the Bridge function supports a capabilities list.
5	P66CAP	R	1	66 MHz Capable. Reads as 1 to indicate that the primary bus of the Bridge function is 66MHz capable. Meaningful only when the SG2010 is a root.
6	RES	R	0	Reserved
7	PFBBC	R	1	Fast Back-to-Back Capable. Reads as 1 to indicate that the Bridge function is capable of accepting a fast back-to-back transaction on the primary bus when they are not to the same target. Meaningful only when the SG2010 is a root.

Bit	Name	Access	Reset Value	Description
8	PMPERR	R/W1TC	0	Master Data Parity Error. Set by the Bridge function when the Primary Parity Error Response bit is set, and the SG2010 detects PERR_L on a write or asserts PERR_L on a read (The SG2010 is the bus master). Set only when the SG2010 is a root.
10:9	PDEVTIM	R	01b	DEVSEL# Timing. Indicates that the Bridge function uses medium timing on the primary bus. Meaningful only when the SG2010 is a root.
11	PSTA	R/W1TC	0	Signaled Target Abort. Set by the Bridge function when it returns a target abort on the primary bus.Set only when the SG2010 is a root.
12	PRTA	R/W1TC	0	Received Target Abort. Set by the Bridge function when it receives a target abort on the primary bus. Set only when the SG2010 is a root.
13	PRMA	R/W1TC	0	Received Master Abort. Set by the Bridge function when it detects a master abort in response to a an upstream transaction. Set only when the SG2010 is a root.
14	PSSERR	R/W1TC	0	Signaled System Error. Set by the Bridge function when it asserts SERR_L on the primary bus.
15	PDPE	R/W1TC	0	Detected Parity Error. Set by the Bridge function when it detects a data or address parity error on the primary bus. Set only when the SG2010 is a root.

4.7.1.5 Revision ID

Bridge Cfg Byte Offset	08h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	REVID	R	H/W	Identifies the silicon revision of this device.

4.7.1.6 Class Code

Bridge Cfg Byte Offset	09h:0Bh
Size	3 bytes

Bit	Name	Access	Reset Value	Description
7:0	PROGIF	R	0	Identifies the programming interface of the device. Reads as 0 to indicate that there is no programming interface.
15:8	SUBCL	R	04h	Identifies the sub-class of the device. Reads as 04h to indicate that this is a PCI-to-PCI bridge device.
23:16	BASECL	R	06h	Identifies the base class of the device. Reads as 06h to indicate that this is a bridge device.

4.7.1.7 Cache Line Size

Bridge Cfg Byte Offset	0Ch
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	CLS	R/W	0	Indicates the cache line size in Dwords used by the SG2010 for generating MWI transactions, disconnecting write transactions, and prefetching read data. Cache line sizes of 8, 16, and 32 Dwords are supported. All other values default to eight Dwords, but the SG2010 does not use the MWI command in these cases.

4.7.1.8 Master Latency Timer

Bridge Cfg Byte Offset	0Dh
Size	1 byte

Bit	Name	Access	Reset Value	Description
2:0	RES	R	0	Reserved
7:3	MLT	R/W	0	Controls the number of PCI clock cycles that the SG2010 may use the primary PCI bus as a bus master. When the timer expires and SG2010's grant is not asserted, the SG2010 must relinquish the PCI bus after the next data phase (or at the next cache line boundary if an MWI). If the timer is set to 0, it is expired when the transaction is initiated on the PCI bus – up to two data phases can occur without grant. Meaningful only when the SG2010 is the root.

4.7.1.9 Header Type

Bridge Cfg Byte Offset	0Eh
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	HDRTYP	R	81h (root) 01h (leaf)	Bit [0] of the header type reads as 1 to indicate that this is a Bridge header. Bit [7] reads as 1 when the SG2010 is the root to indicate that the Bridge is functioning as one function in a multifunction device. Bit [7] reads as 0 when the SG2010 is a leaf to indicate that the Bridge is functioning as a single function device.

4.7.2 PCI Address and Secondary Bus Registers

4.7.2.1 Primary Bus Number

Bridge Cfg Byte Offset	18h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	PBUS	R/W	0h	Identifies the bus number of the primary PCI bus. Used for upstream decoding of Type1 configuration transactions to determine whether they are to be translated to special cycles.

4.7.2.2 Secondary Bus Number

Bridge Cfg Byte Offset	19h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	SBUS	R/W	0h	Identifies the bus number of the secondary PCI bus. Used to define the base bus number of a window used for bus number decoding of Type1 configuration transactions. Additionally, used for decoding of downstream transactions to determine whether they are translated to Type0.

4.7.2.3 Subordinate Bus Number

Bridge Cfg Byte Offset	1Ah
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	SUBBUS	R/W	0h	Identifies the bus number of the subordinate PCI bus. Used to define the upper limit (inclusive) of a window for decoding bus numbers of Type1 configuration transactions.

4.7.2.4 Secondary Latency Timer

Bridge Cfg Byte Offset	1Bh
Size	1 byte

Bit	Name	Access	Reset Value	Description
2:0	RES	R	0	Reserved
7:3	SMLT	R/W	0h	Controls the number of PCI clock cycles that the SG2010 may use the secondary PCI bus as a bus master. When the timer expires and SG2010’s grant is not asserted, the SG2010 must relinquish the PCI bus after the next data phase (or at the next cache line boundary if an MWI). If the timer is set to 0, it is expired when the transaction is initiated on the PCI bus – up to two data phases can occur without grant. Meaningful only when the SG2010 is a leaf.

4.7.2.5 I/O Base

Bridge Cfg Byte Offset	1Ch
Size	1 byte

Bit	Name	Access	Reset Value	Description
0	IO32	R	1	Reads as 1 to indicate that the SG2010 supports a 32-bit I/O address range as a PCI-to-PCI bridge.
3:1	RES	R	0	Reserved.
7:4	IOBASE	R/W	0	Defines address bits [15:12] of the low end of the downstream I/O address range. Bits [11:0] are assumed to be 0, giving a minimum size and alignment granularity of 4KB.

4.7.2.6 I/O Limit

Bridge Cfg Byte Offset	1Dh
Size	1 byte

Bit	Name	Access	Reset Value	Description
0	IO32	R	1	Reads as 1 to indicate that the SG2010 supports a 32-bit I/O address range as a PCI-to-PCI bridge.
3:1	RES	R	0	Reserved.
7:4	IOLIMIT	R/W	0	Defines address bits [15:12] of the high end of the downstream I/O address range. Bits [11:0] are assumed to be 0, giving a minimum size and alignment granularity of 4KB.

4.7.2.7 Secondary Status

Bridge Cfg Byte Offset	1Eh:1Fh
Size	2 bytes

Bit	Name	Access	Reset Value	Description
4:0	RES	R	0	Reserved
5	S66CAP	R	1	66 MHz Capable. Reads as 1 to indicate that the secondary bus of the Bridge function is 66MHz capable. Meaningful only when the SG2010 is a leaf.
6	RES	R	0	Reserved
7	SFBBC	R	1	Fast Back-to-Back Capable. Reads as 1 to indicate that the Bridge function is capable of accepting a fast back-to-back transaction on the secondary bus when they are not to the same target. Meaningful only when the SG2010 is a leaf.
8	SMPERR	R/W1TC	0	Master Data Parity Error. Set by the Bridge function when the Secondary Parity Error Response bit is set, and the SG2010 detects PERR_L on a write or asserts PERR_L on a read (The SG2010 is the bus master). Set only when the SG2010 is a leaf.

Bit	Name	Access	Reset Value	Description
10:9	SDEVTIM	R	01b	DEVSEL# Timing. Indicates that the Bridge function uses medium timing on the secondary bus. Meaningful only when the SG2010 is a leaf.
11	SSTA	R/W1TC	0	Signaled Target Abort. Set by the Bridge function when it returns a target abort on the secondary bus. Set only when the SG2010 is a leaf.
12	SRTA	R/W1TC	0	Received Target Abort. Set by the Bridge function when it receives a target abort on the secondary bus. Set only when the SG2010 is a leaf.
13	SRMA	R/W1TC	0	Received Master Abort. Set by the Bridge function when it detects a master abort in response to a downstream transaction or frame.
14	SRSERR	R/W1TC	0	Received System Error. Set by the Bridge function when it detects SERR_L asserted on the secondary bus. Set only when the SG2010 is a leaf.
15	SDPE	R/W1TC	0	Detected Parity Error. Set by the Bridge function when it detects a data or address parity error on the secondary bus. Set only when the SG2010 is a leaf.

4.7.2.8 Memory Base

Bridge Cfg Byte Offset	20h:21h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
3:0	RES	R	0	Reserved.
15:4	MEMBASE	R/W	0	Defines address bits [31:20] of the low end of the down-stream memory address range. Bits [19:0] are assumed to be 0, giving a minimum size and alignment granularity of 1MB.

4.7.2.9 Memory Limit

Bridge Cfg Byte Offset	22h:23h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
3:0	RES	R	0	Reserved.
15:4	MEMLIMIT	R/W	0	Defines address bits [31:20] of the high end of the down-stream memory address range. Bits [19:0] are assumed to be 0, giving a minimum size and alignment granular-ity of 1MB.

4.7.2.10 Prefetchable Memory Base

Bridge Cfg Byte Offset	24h:25h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
0	MEM64	R	1	Reads as 1 to indicate that the SG2010 supports a 64-bit prefetchable memory address range as a PCI-to-PCI bridge.
3:1	RES	R	0	Reserved.
15:4	PFMBASE	R/W	0	Defines address bits [31:20] of the low end of the downstream prefetchable memory address range. Bits [19:0] are assumed to be 0, giving a minimum size and alignment granularity of 1MB.

4.7.2.11 Prefetchable Memory Limit

Bridge Cfg Byte Offset	26h:27h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
0	MEM64	R	1	Reads as 1 to indicate that the SG2010 supports a 64-bit prefetchable memory address range as a PCI-to-PCI bridge.
3:1	RES	R	0	Reserved.
15:4	PFMLIMIT	R/W	0	Defines address bits [31:20] of the high end of the downstream prefetchable memory address range. Bits [19:0] of the address are assumed to be 0, giving a minimum size and alignment granularity of 1MB.

4.7.2.12 Prefetchable Memory Base Upper 32 Bits

Bridge Cfg Byte Offset	28h:2Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	PFMB64	R/W	0	Defines address bits [63:32] of the low end of the downstream prefetchable memory address range.

4.7.2.13 Prefetchable Memory Limit Upper 32 Bits

Bridge Cfg Byte Offset	2Ch:2Fh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	PFML64	R/W	0	Defines address bits [63:32] of the high end of the downstream prefetchable memory address range.

4.7.2.14 I/O Base Upper 16 Bits

Bridge Cfg Byte Offset	30h:31h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
15:0	IOB32	R/W	0	Defines address bits [31:16] of the low end of the downstream I/O address range.

4.7.2.15 I/O Limit Upper 16 Bits

Bridge Cfg Byte Offset	32h:33h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
15:0	IOL32	R/W	0	Defines address bits [31:16] of the high end of the downstream I/O address range.

4.7.3 Other PCI Registers

4.7.3.1 ECP

Bridge Cfg Byte Offset	34h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	ECP	R	44h	Returns the configuration offset of the first ECP function, which is the Power Management ECP function at offset 44h.

4.7.3.2 Interrupt Line

Bridge Cfg Byte Offset	3Ch
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	INTLINE	R/W	0	Interrupt line register. Initialization code should write this to FFh; the Bridge function does not support an interrupt pin.

4.7.3.3 Interrupt Pin

Bridge Cfg Byte Offset	3Dh
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	INTPIN	R	0	Reads as 0 to indicate that the Bridge does not support an interrupt pin.

4.7.3.4 Bridge Control

Bridge Cfg Byte Offset	3Eh:3Fh
Size	2 bytes

Bit	Name	Access	Reset Value	Description
0	SPER	R/W	0	Parity Error Response for the secondary PCI bus. When: 0 The SG2010 detects, but does not report, parity errors (PERR_L assertion, SERR_L assertion, setting the Master Data Parity Error bit) on the secondary PCI bus. 1 The SG2010 is enabled to report parity errors detected on the secondary PCI bus.
1	SERRFE	R/W`	0	SERR# Forward Enable. When: 0 The SG2010 ignores SERR_L asserted on the secondary PCI bus when in default mode, and a leaf with Bridge function enabled. 1 When in default mode, the SG2010 as a leaf with Bridge function enabled, forwards SERR_L assertions detected on the secondary bus upstream if the SERR# Enable bit is also set. Not meaningful when the SG2010 is a root.
2	ISAENA	R/W	0	ISA Enable. When: 0 The SG2010 does not perform ISA address filtering. 1 The SG2010 performs ISA address filtering in the I/O Base and Limit Range.
3	VGAENA	R/W	0	VGA Enable. When: 0 The SG2010 does not perform VGA address decoding. 1 The SG2010 decodes VGA addresses for downstream forwarding, and inverse decodes them for upstream forwarding.
4	VGA16	R/W or R	0	VGA 16-bit decode. Controls aliasing for VGA transactions. The VGA 16-bit Decode Support bit (Chip Control register, Gateway Configuration space) must be set to a 1 to activate this bit as read/write, otherwise it is read-only as 0. When: 0 A 10-bit address decode is performed and address bits [15:10] may be any value to forward VGA transactions downstream. 1 A full 16-bit address decode is performed and bits [15:10] must be 0.

Bit	Name	Access	Reset Value	Description
5	MAMODE	R/W	0	Master Abort Mode. In conjunction with the PCI Target Response Mode bit in the Gateway Chip Control register, controls SG2010's response on the PCI bus to a variety of failures. 0 When this bit and the PCI Target Response bit are both 0, a benign response is returned; that is, write data is discarded and reads return FFFFFFFFh. 1 When this bit or the PCI Target Response bit is a 1, an error response is returned – a target abort is returned for delayed transactions and a system error event is signaled for writes without acknowledges. Table 3–8 lists the failures where this bit is used.
6	SECRST	R/W	0	Secondary Reset. When: 0 Clears secondary reset. The SG2010 deasserts RSTO_L. 1 A secondary PCI reset is generated. The SG2010 generates an address routed reset comma, clears its Port Map Tables, and asserts RSTO_L.
7	FBBENS	R/W	0	Fast Back-to-back Enable for the secondary PCI bus. When: 0 The SG2010 does not perform fast back-to-back transactions on the secondary PCI bus. 1 The SG2010 is enabled to generate fast back-to-back transactions on the secondary PCI bus. Not meaningful when the SG2010 is a root.
8	PDISC	R/W	0	Primary Discard Timer. When: 0 Delayed transactions are discarded 2^{15} PCI clock cycles after delayed transaction status is ready to return and the initiator has not re-attempted the transaction on the primary PCI bus. 1 Delayed transactions are discarded 2^{10} PCI clock cycles after delayed transaction status is ready to return and the initiator has not re-attempted the transaction. Not meaningful when the SG2010 is a leaf.
9	SDISC	R/W	0	Secondary Discard Timer. When: 0 Delayed transactions are discarded 2^{15} PCI clock cycles after delayed transaction status is ready to return and the initiator has not re-attempted the transaction on the secondary bus. 1 Delayed transactions are discarded 2^{10} PCI clock cycles after delayed transaction status is ready to return and the initiator has not re-attempted the transaction. Not meaningful when the SG2010 is a root.

Bit	Name	Access	Reset Value	Description
10	DISCSTAT	R/W1TC	0	Master Time-out Discard Timer Status. Set by the SG2010 when a delayed transaction is discarded due to a master not reattempting the transaction. Software writes 1 to this bit to clear it.
11	DISCSERR	R/W	0	Master Time-out Discard Timer SERR# Enable. When: 0 The Discard Timer Status event does not cause a SERR# assertion in default mode. 1 The Discard Timer Status event causes a SERR# assertion to the root in default mode (if the SERR# Enable bit is set).
15:12	RES	R	0	Reserved.

4.7.4 Bridge Power Management Registers

The Bridge and Gateway functions have separate power management functions. The power management functions are not dual-mapped versions of the same function.

4.7.4.1 Power Management ECP ID

Bridge Cfg Byte Offset	44h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	PMECPID	R	01h	Extended Capabilities ID for the power management function. Must read as 01h.

4.7.4.2 Power Management Next Pointer

Bridge Cfg Byte Offset	45h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	PMNPTR	R	4Ch	Contains the configuration offset of the next ECP function, which is the Slot Numbering ECP function at offset 4Ch.

4.7.4.3 Power Management Capabilities

Bridge Cfg Byte Offset	46h:47h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
2:0	PMVER	R	2h	Reads as 010b to indicate that this device complies with version 1.1 of the *PCI Power Management Specification.*
3	PMECLK	R	0	PCI clock required for PME_L. This function does not support PME_L and this bits reads as 0.
4	RES	R	0	Reserved.
5	DSI	R	0	Device Specific Initialization required. Reads as 0 to indicate that device specific initialization is not required following a transition to D0.
8:6	AUXCURR	R	0	3.3V auxiliary current requirements. This function does not support PME_L and these bits read as 0.
9	D1_SUPP	R	0	D1 Support. Reads as 0 to indicate that this function does not support the D1 power state.
10	D2_SUPP	R	0	D2 Support. Reads as 0 to indicate that this function does not support the D2 power state.
15:11	PME_SUPP	R	0	PME# Support. Reads as 0 to indicate that PME_L assertion is not supported in any power state.

4.7.4.4 Power Management Control and Status

Bridge Cfg Byte Offset	48h:49h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
1:0	PWRST	R/W	0	Indicates the current power state of this function and controls transitions to a new power state. Only D0 (00b) and D3 (11b) are supported, and only writes to these states are allowed. The SG2010 remains in the current power state if a write is attempted to either D1 or D2.
7:2	RES	R	0	Reserved.
8	PME_EN	R	0	PME# Enable. Reads as 0 because PME_L is not supported by this function.
12:9	DATA_SEL	R	0	Data Select Index. Reads as 0 because the Data register is not supported.
14:13	SCALE	R	0	Data Scale. Reads as 0 because the Data register is not supported.
15	PME_ST	R	0	PME# Status. Reads as 0 because this function does not support PME_L.

4.7.4.5 Power Management PCI-to-PCI Bridge Support

Bridge Cfg Byte Offset	4Ah
Size	1 byte

Bit	Name	Access	Reset Value	Description
5:0	RES	R	0	Reserved.
6	B2_B3	R	0	B2/B3 support. Always reads as zero because the SG2010 does not support PCI clock outputs.
7	BPCC_EN	R	0	Clock control enable. Always reads as zero because the SG2010 does not support PCI clock outputs.

4.7.4.6 Power Management Data

Bridge Cfg Byte Offset	4Bh
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	PMDAT	R	0	Data register. Not supported, reads as 0.

4.7.5 Slot Numbering Registers

To use this function, these registers must be loaded through serial ROM preload.

4.7.5.1 Slot Numbering ECP ID

Bridge Cfg Byte Offset	4Ch
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	SNECPID	R	04h	Extended Capabilities ID for the slot numbering function. Must read as 04h.

4.7.5.2 Slot Numbering Next Pointer

Bridge Cfg Byte Offset	4Dh
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	SNNEXTPTR	R	58h	Contains the configuration offset of the next ECP function, which is CompactPCI Hot Swap at offset 58h.

4.7.5.3 Slot Numbering Expansion Slot

Bridge Cfg Byte Offset	4Eh
Size	1 byte

Bit	Name	Access	Reset Value	Description
4:0	EXPSLT	R	0	Indicates the number of expansion slots located on SG2010's secondary bus.
5	CHASSIS	R	0	When set to 1 through the serial ROM, indicates that this is the first bridge in an expansion chassis.
7:6	RES	R	0	Reserved. Serial pre-load data should contain 0's (not hardware restricted).

4.7.5.4 Slot Numbering Chassis Number

Bridge Cfg Byte Offset	4Fh
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	CHASNUM	R/W	0	Contains the chassis number for the secondary bus.

4.7.6 Vital Product Data (VPD) Registers

These registers are a dual-mapped copy of the VPD registers in the Gateway configuration registers. There is only a single VPD function.

4.7.6.1 VPD ECP ID

Bridge Cfg Byte Offset	50h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	VPDECPID	R	3h	Extended Capabilities ID for the VPD function. Must read as 3h.

4.7.6.2 VPD Next Pointer

Bridge Cfg Byte Offset	51h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	VPDNPTR	R	0	Contains the configuration offset of the next ECP function. VPD is the last function in the list, and this register returns 0 when read.

4.7.6.3 VPD Address

Bridge Cfg Byte Offset	52h:53h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
7:0	VPDADDR	R/W	0	Dword aligned byte-address offset of the 256-byte VPD address region to be accessed (bits [1:0] are ignored and assumed to be 00b).
14:8	RES	R	0	Reserved
15	VPDFLAG	R/W	0	VPD operation/status bit. When written with: 0 The SG2010 reads four bytes of data from the VPD address location written to bits [7:0]. When the read is complete and data is available, the SG2010 sets this bit to 1. 1 The SG2010 writes four bytes of data contained in the VPD data register to the VPD address location written to bits [7:0]. When the write is complete, the SG2010 sets this bit to 0.

4.7.6.4 VPD Data

Bridge Cfg Byte Offset	54h:57h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	VPDDATA	R/W	0	Contains four bytes of VPD data. On a VPD read, the SG2010 places read data in this register. Before a VPD write operation, software must write the data to be stored to this register.

4.7.7 CompactPCI Hot Swap Registers

Only the assertion of LRST_L or RST_L resets the hot swap register bits; they cannot be reset by other mechanisms.

4.7.7.1 Hot Swap ECP ID

Bridge Cfg Byte Offset	58h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	HSECPID	R	6h	Extended Capabilities ID for the hot swap function. Must read as 06h.

4.7.7.2 Hot Swap Next Pointer

Bridge Cfg Byte Offset	59h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	HSNPTR	R	50	Contains the configuration offset of the next ECP function, Which is VPD at offset 50h.

4.7.7.3 Hot Swap Control

Bridge Cfg Byte Offset	5Ah
Size	1 byte

Bit	Name	Access	Reset Value	Description
0	DHE	R/W	0	Device Hiding Enable. Implemented when PI = 1. When this bit is: 0 Device hiding is disabled. 1 Device hiding is enabled. If PI = 0, this bit is read-only and returns 0.
1	EIM	R/W	0	ENUM_L interrupt mask. When: 0 Enables the assertion of ENUM_L when INS or EXT are set. 1 Masks the assertion of ENUM_L.
2	PIE	R	H/W	Pending Insertion/Extraction. Implemented when PI = 1. Reads as: 0 When the SG2010 is in the installed state. 1 When insertion or extraction is in progress.
3	LOO	R/W	0	LED on/off. When the LED is under software control and LOO is: 0 The HS_LED signal is deasserted and the LED is off. 1 The HS_LED signal is asserted and the LED is illuminated.
5:4	PI	R	1	Programming Interface. Indicates the CompactPCI Hot Swap programming interface revision. By default, this bit is 1, but can be cleared to 0 through serial preload. When: 0 Only the EIM, LOO, EXT, and INS bits are implemented; the remaining bits in this register read as 0. Device hiding and software-initiated extraction are not implemented. 1 All bits are implemented, and device hiding and software-initiated extraction are supported.
6	EXT	R/W1TC	0	Pending extraction status bit. Set by the SG2010 when the ejector handle is opened (LSTAT = 1) and the board is in the Installed state. Cleared when software writes 1 to this location. Writing 0 has no effect.
7	INS	R/W1TC	0	Board inserted status bit. Set by the SG2010 when the device is ready for configuration, after ejector handle has been closed (LSTAT = 0), reset has been deasserted, and any local pre-initialization has been completed. Cleared when software writes 1 to this location. Writing 0 has no effect.

4.8 Gateway Configuration Registers

This section contains the configuration register descriptions for the Gateway device. The Gateway configuration space also includes all the device-specific configuration bits, whether the register applies to Bridge functionality, Gateway functionality, or both.

4.8.1 PCI Header Registers

4.8.1.1 Vendor ID

Gateway Cfg Byte Offset	00h:01h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
15:0	VENDORID	R	9902h	Identifies the vendor of this device as StarGen. Returns 9902h when read.

4.8.1.2 Device ID

Gateway Cfg Byte Offset	02h:03h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
15:0	DEVID	R	02h	PCI device identification number for the Gateway. Returns 02h when read.

4.8.1.3 Command

Gateway Cfg Byte Offset	04h:05h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
0	GIO	R/W	0	I/O Enable for the Gateway function’s PCI bus. When: 0 The Gateway function does not respond to I/O transactions on the PCI bus. 1 The Gateway function is enabled to respond to I/O transactions on the PCI bus.
1	GMEM	R/W	0	Memory Enable for the Gateway function. When 0 The Gateway function does not respond to memory transactions on the PCI bus. 1 The Gateway function is enabled to respond to memory transactions on the PCI bus.
2	GBM	R/W	0	Bus Master Enable for the Gateway function. When: 0 The Gateway function does not initiate memory or I/O transactions on the PCI bus. 1 The Gateway function is enabled to initiate memory or I/O transactions on the PCI bus.

Bit	Name	Access	Reset Value	Description
3	GSC	R	0	Special Cycles. Reads as 0 because the SG2010 does not monitor special cycles on the PCI bus.
4	GMWI	R/W	0	MWI Enable for the Gateway function. When: 0 The Gateway does not initiate Memory Write and Invalidate transactions on the PCI bus. 1 The Gateway is enabled to initiate Memory Write and Invalidate transactions.
5	VGASN	R	0	VGA Snoop. Reads as 0 because the SG2010 does not support VGA snooping.
6	GPER	R/W	0	Parity Error Response for the Gateway function. When: 0 The Gateway detects but does not report parity errors on the PCI bus (PERR_L assertion, SERR_L assertion, setting the Master Data Parity Error bit). 1 The Gateway is enabled to report parity errors detected on the PCI bus.
7	STEP	R	0	Stepping Control. Reads as 0 because the SG2010 does not perform stepping.
8	GSERREN	R/W	0	SERR# Enable for the Gateway function. When: 0 And the Bridge SERREN is also 0, the Gateway does not assert SERR_L. 1 The Gateway is enabled to assert SERR_L on the PCI bus.
9	GFBBEN	R/W	0	Fast Back-to-back Enable for the Gateway function. When: 0 The Gateway does not perform fast back-to-back transactions on the PCI bus. 1 The Gateway is enabled to generate fast back-to-back transactions on the PCI bus.
15:10	RES	R	0	Reserved

4.8.1.4 Status

Gateway Cfg Byte Offset	06h:07h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
3:0	RES	R	0	Reserved
4	GCAP	R	1	Capabilities List. Reads as 1 to indicate that the Gateway function supports a capabilities list.
5	G66CAP	R	1	66 MHz Capable. Reads as 1 to indicate that the Gateway function is 66MHz capable.
6	RES	R	0	Reserved
7	GFBBC	R	1	Fast Back-to-Back Capable. Reads as 1 to indicate that the Gateway function is capable of accepting a fast back-to-back transaction when they are not to the same target.

Bit	Name	Access	Reset Value	Description
8	GMPERR	R/W1TC	0	Master Data Parity Error. Set by the Gateway function when the Parity Error Response bit is set, and the SG2010 detects PERR_L on a write or asserts PERR_L on a read (The SG2010 is the bus master).
10:9	GDEVTIM	R	01b	DEVSEL# Timing. Indicates that the Gateway function uses medium timing on the PCI bus.
11	GSTA	R/W1TC	0	Signaled Target Abort. Set by the Gateway function when it returns a target abort on the PCI bus.
12	GRTA	R/W1TC	0	Received Target Abort. Set by the Gateway function when it receives a target abort on the PCI bus.
13	GRMA	R/W1TC	0	Received Master Abort. Set by the Gateway function when it detects a master abort in response to a transaction it initiated on the PCI bus.
14	GSSERR	R/W1TC	0	Signaled System Error. Set by the Gateway function when it asserts SERR_L on the PCI bus.
15	GDPE	R/W1TC	0	Detected Parity Error. Set by the Gateway function when it detects a data or address parity error on the PCI bus.

4.8.1.5 Revision ID

Gateway Cfg Byte Offset	08h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	REVID	R	H/W	Identifies the silicon revision of this device.

4.8.1.6 Class Code

Gateway Cfg Byte Offset	09h:0Bh
Size	3 bytes

Bit	Name	Access	Reset Value	Description
7:0	PROGIF	R	0	Identifies the programming interface of the device. Reads as 0 to indicate that there is no programming interface.
15:8	SUBCL	R	80h	Identifies the subclass of the device. Reads as 80h to indicate that this is an “other” bridge device.
23:16	BASECL	R	06h	Identifies the base class of the device. Reads as 06h to indicate that this is a bridge device.

4.8.1.7 Cache Line Size

Gateway Cfg Byte Offset	0Ch
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	CLS	R/W	0	Indicates the cache line size in Dwords used by the Gateway function for generating MWI transactions, disconnecting write transactions as a target, and determining speculative read prefetch amounts. Cache line sizes of 8, 16, and 32 Dwords are supported. All other values default to eight Dwords, but in this case the SG2010 does not use the MWI command.

4.8.1.8 Master Latency Timer

Gateway Cfg Byte Offset	0Dh
Size	1 byte

Bit	Name	Access	Reset Value	Description
2:0	RES	R	0	Reserved
7:3	MLT	R/W	0	Controls the number of PCI clock cycles that the SG2010 may use the PCI bus as a bus master. When the timer expires and the SG2010 grant is not asserted, the SG2010 must relinquish the PCI bus after the next data phase (or at the next cache line boundary if an MWI). If the timer is set to 0, it is expired when the SG2010 initiates the transaction on the PCI bus – up to two data phases can occur without grant.

4.8.1.9 Header Type

Gateway Cfg Byte Offset	0Eh
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	HDRTYPE	R	00h	Reads as 0 to indicate that a type 0 configuration header is used.

4.8.2 PCI Address Registers

Gateway address registers use standard BARs defining PCI address ranges for CSR access and for forwarding and translation to path-routed or multicast frames.

4.8.2.1 BAR0

Gateway Cfg Byte Offset	10h:13h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
0	TYPESEL	R	0	Specifies whether this address range is to be mapped into memory or I/O space. Reads as 0 to indicate that this is a memory BAR.
2:1	TYPE	R	0	Size and location of space selected. Reads as 00b to indicate that this is a 32-bit BAR that can be mapped anywhere in 32-bit address space.
3	PF	R	0	Reads as 0 to indicate that this address region is not prefetchable.
14:4	RES	R	0	Reserved.
31:15	BASE	R/W	0	Specifies the size of the address range (power-of-2 granularity) and the base address value. This size is fixed at 32KB to map SG2010 CSRs into PCI memory space.

4.8.2.2 BAR1

Gateway Cfg Byte Offset	14h:17h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
0	TYPESEL	R	1	Specifies whether this address range is to be mapped into memory or I/O space. Reads as 1 to indicate that this is an I/O BAR.
6:1	RES	R	0	Reserved.
31:7	BASE	R/W	0	Specifies the size of the address range (power-of-2 granularity) and the base address value. This BAR requests 128 bytes of I/O space to map SG2010 CSRs.

4.8.2.3 BAR2

Gateway Cfg Byte Offset	18h:1Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
0	TYPESEL	R	0	Specifies whether this address range is to be mapped into memory or I/O space. Reads as 0 to indicate that this is a memory BAR.
2:1	TYPE	R	0	Size and location of space selected. This BAR can either be programmed to be a 32-bit BAR (00b) or the lower 32 bits of a 64-bit BAR (10b).
3	PF	R	0	When: 0 Indicates that this space is non-prefetchable. 1 Indicates that this space is prefetchable.

Bit	Name	Access	Reset Value	Description
18:4	RES	R	0	Reserved.
31:19	BASE	R/W	0	Specifies the size of the address range (power-of-2 granularity) and the base address value. This address range is used to decode and forward path-routed or multicast frames. The corresponding setup register determines the size requested. The minimum size is 512KB and the maximum size is 2GB for a 32-bit BAR and 1PB (1 Petabyte or 2^{50} bytes) for a 64-bit BAR.

4.8.2.4 BAR3

This BAR can be either a 32-bit BAR or the upper 32 bits of a 64-bit BAR. The description for a 32-bit bar follows. If a 64-bit BAR, all bits are base address bits.

Gateway Cfg Byte Offset	1Ch:1Fh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
0	TYPESEL	R	0	Specifies whether this address range is to be mapped into memory or I/O space. Reads as 0 to indicate that this is a memory BAR.
2:1	TYPE	R	0	Size and location of space selected. Reads as 00b to indicate that this is a 32-bit BAR that can be mapped anywhere in 32-bit address space.
3	PF	R	0	When: 0 Indicates that this space is non-prefetchable. 1 Indicates that this space is prefetchable.
18:4	RES	R	0	Reserved.
31:19	BASE	R/W	0	Specifies the size of the address range (power-of-2 granularity) and the base address value. This address range is used to decode and forward path-routed or multicast frames. The corresponding setup register determines the size requested. The minimum size is 512KB and the maximum size is 2GB for a 32-bit BAR and 1PB (1 Petabyte or 2^{50} bytes) for a 64-bit BAR.

4.8.2.5 BAR4

Gateway Cfg Byte Offset	20h:23h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
0	TYPESEL	R	0	Specifies whether this address range is to be mapped into memory or I/O space. Reads as 0 to indicate that this is a memory BAR.
2:1	TYPE	R	0	Size and location of space selected. This BAR can either be programmed to be a 32-bit BAR (00b) or the lower 32 bits of a 64-bit BAR (10b).
3	PF	R	0	When: 0 Indicates that this space is non-prefetchable. 1 Indicates that this space is prefetchable.
18:4	RES	R	0	Reserved.
31:19	BASE	R/W	0	Specifies the size of the address range (power-of-2 granularity) and the base address value. This address range is used to decode and forward path-routed or multicast frames. The corresponding setup register determines the size requested. The minimum size is 512KB and the maximum size is 2GB for a 32-bit BAR and 1PB (1 Petabyte or 2^{50} bytes) for a 64-bit BAR.

4.8.2.6 BAR5

This BAR can be either a 32-bit BAR or the upper 32 bits of a 64-bit BAR. The description for a 32-bit bar follows. If a 64-bit BAR, all bits are base address bits.

Gateway Cfg Byte Offset	24h:27h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
0	TYPESEL	R	0	Specifies whether this address range is to be mapped into memory or I/O space. Reads as 0 to indicate that this is a memory BAR.
2:1	TYPE	R	0	Size and location of space selected. Reads as 00b to indicate that this is a 32-bit BAR that can be mapped anywhere in 32-bit address space.
3	PF	R	0	When: 0 Indicates that this space is non-prefetchable. 1 Indicates that this space is prefetchable.

Bit	Name	Access	Reset Value	Description
18:4	RES	R	0	Reserved.
31:19	BASE	R/W	0	Specifies the size of the address range (power-of-2 granularity) and the base address value. This address range is used to decode and forward path-routed or multicast frames. The corresponding setup register determines the size requested. The minimum size is 512KB and the maximum size is 2GB for a 32-bit BAR and 1PB (1 Petabyte or 2^{50} bytes) for a 64-bit BAR.

4.8.3 Other PCI Registers

4.8.3.1 Subsystem Vendor ID

Gateway Cfg Byte Offset	2Ch:2Dh
Size	2 bytes

Bit	Name	Access	Reset Value	Description
15:0	SVENDOR	R	0	Returns the Vendor ID for the subsystem containing this device. This register can be preloaded by serial ROM

4.8.3.2 Subsystem ID

Gateway Cfg Byte Offset	2Eh:2Fh
Size	2 bytes

Bit	Name	Access	Reset Value	Description
15:0	SSYSID	R	0	Returns the ID for the subsystem containing this device. This register can be preloaded by serial ROM.

4.8.3.3 Expansion ROM BAR

Gateway Cfg Byte Offset	30h:33h
Size	4 bytes

Bit	Name	Access	Reset Value	Description

0	DECENA	R/W	0	Address decode enable for this BAR. When: 0 This address range is disabled for address decode, and the ROM cannot be read through this mechanism 1 The SG2010 decodes incoming transactions against this address range for possible ROM read access (as long as the Enable bit in the Expansion ROM BAR Setup register is also set).
11:1	RES	R	0	Reserved.
31:12	BASE	R/W	0	Specifies the size of the address range (power-of-2 granularity) and the base address value. This address range is used for accessing the parallel ROM. The Expansion ROM Setup register determines whether this BAR is present and the size requested (maximum 16MB).

4.8.3.4 ECP

Gateway Cfg Byte Offset	34h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	ECP	R	ACh	Returns the configuration offset of the first ECP function, which is the Power Management function at offset ACh.

4.8.3.5 Interrupt Line

Gateway Cfg Byte Offset	3Ch
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	INTLINE	R/W	0	Indicates interrupt routing information for the corresponding interface.

4.8.3.6 Interrupt Pin

Gateway Cfg Byte Offset	3Dh
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	INTPIN	R	1	Indicates which interrupt pin this function uses. Reads as 1 to indicate that the Gateway uses INTA_L.

4.8.3.7 Minimum Grant (MIN_GNT)

Gateway Cfg Byte Offset	3Eh
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	MINGNT	R	0	Indicates the burst length required in 0.25µs units. This register can be preloaded by serial ROM.

4.8.3.8 Maximum Latency (MAX_LAT)

Gateway Cfg Byte Offset	3Fh
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	MAXLAT	R	0	Indicates the frequency of bus access required in 0.25µs units. This register can be preloaded by serial ROM.

4.8.4 Software Generated Transaction (SGT) Registers

4.8.4.1 SGT Configuration Address

Gateway Cfg Byte Offset	40h:43h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	CFGADDR	R/W	0	Specifies the address to be used for a software-generated configuration transaction on the PCI bus. The address in this register must be valid before the SGT Configuration Data register is accessed, and is driven on the PCI bus as written.

4.8.4.2 SGT Configuration Data

Gateway Cfg Byte Offset	44h:47h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	CFGDATA	R	0	When the Configuration SGT Enable in the Chip Control register is set to 1, and this register is accessed from the fabric, the SG2010 initiates a configuration transaction on the PCI bus. If the register is written, a configuration write transaction is generated using the write data. When the write is completed, a write acknowledge is returned to the initiator. If the register is read, a configuration read transaction is generated. When the read data is returned, it is placed in a completion frame for the initiator. The byte enables of the PCI transaction correspond to the byte mask used for the SGT Configuration Data register access. The address used is the value in the SGT Configuration Address register. If the Configuration SGT Enable bit is 0, or the access is from the PCI bus, write data is discarded and reads return 0.

4.8.4.3 SGT I/O Address

Gateway Cfg Byte Offset	48h:4Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	IOADDR	R/W	0	Specifies the address to be used for a software-generated I/O transaction on the PCI bus. The address in this register must be valid before the SGT I/O Data register is accessed, and is driven on the PCI bus as written.

4.8.4.4 SGT I/O Data

Gateway Cfg Byte Offset	4Ch:4Fh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	IODATA	R	0	When the I/O SGT Enable in the Chip Control register is set to 1, and this register is accessed from the fabric, the SG2010 initiates an I/O transaction on the PCI bus. If the register is written, an I/O write transaction is generated using the write data. When the write is completed, an acknowledge is returned to the initiator. If the register is read, an I/O read transaction is generated. When the read data is returned, it is placed in a completion frame for the initiator. The byte enables of the PCI transaction correspond to the byte mask used for the SGT I/O Data register access. The address used is the value in the SGT I/O Address register. If the I/O SGT Enable bit is 0, or the access is from the PCI bus, write data is discarded and reads return an undetermined value.

4.8.5 ROM Control Registers

4.8.5.1 ROM Setup

Gateway Cfg Byte Offset	58h:5Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
0	MLTDEV	R/W	0	Multiple Device Mode. Specifies whether the parallel ROM interface is operating in multiple device mode. When: 0 The SG2010 supports only a single parallel ROM attached to the parallel ROM interface and drives a single chip select signal. 1 The SG2010 supports multiple devices on the parallel ROM interface and assumes that upper address bits are externally decoded to generate chip select signals.
2:1	ACCTIME	R/W	0	Access Time. This field sets the amount of time a chip select is asserted for a device attached to the parallel ROM interface. In single device mode, the PR_CS_L pin is used as the chip select. In multiple device mode, the PR_ALE_L signal controls the assertion of chip selects. Possible values are: 00b: 17 or 34 PCI clock cycles in 33MHz or 66MHz mode 01b: 34 or 68 PCI clock cycles in 33MHz or 66MHz mode 10b: 136 or 272 PCI clock cycles in 33MHz or 66MHz mode 11b: 544 or 1088 PCI clock cycles in 33MHz or 66MHz mode
7:3	RES	R	0	Reserved
15:8	STROBE	R/W	7Eh	Strobe mask. These bits specify the read and write strobe timing for the parallel ROM interface. The strobe mask time period starts at the assertion of the chip select (or PR_ALE_L deassertion in multiple device mode) and ends at the deassertion of the chip select. Each bit is one-eighth of the access time. If a bit is: 0 The read or write strobe is deasserted during that time period. 1 The read or write strobe is asserted during that time period. Bit [8] represents the first time period and bit [15] represents the last time period. Reset value is 01111110b.
31:16	RES	R	0	Reserved.

4.8.5.2 ROM Address

Gateway Cfg Byte Offset	5Ch:5Fh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
23:0	ROMADDR	R/W	0	Specifies the byte address offset for a parallel ROM or SROM access. When a serial ROM access or a parallel ROM Dword access is performed, the two LSBs are ignored and treated as 0. When a parallel ROM byte access is specified, the low two LSBs are used.
28:24	RES	R	0	Reserved.
29	PRBYTE	R/W	0	Parallel ROM byte access enable. When: 0 And a parallel ROM operation is selected, an aligned Dword access is performed using all 32 bits in ROM Data[31:0]. (The parallel ROM sequencer actually performs four byte accesses). 1 And a parallel ROM operation is selected, a byte access is performed using data in ROM Data [7:0]. This bit has no effect on serial ROM accesses.
30	S/P	R/W	0	Selects a serial or parallel ROM operation. When the Start/Busy flag is accessed and this bit is written with: 0 A serial ROM operation is performed. 1 A parallel ROM operation is performed.
31	SBF	R/W	0	Start/Busy Flag. Initiates the ROM transaction. When written with: 0 The SG2010 reads four bytes from the ROM address location specified in bits [23:0]. When the read is complete and the data is available in the ROM Data register, the SG2010 sets this bit to 1. 1 The SG2010 writes four bytes to the ROM address specified in bits [23:0]. When the write is complete, the SG2010 sets this bit to 0.

4.8.5.3 ROM Data

Gateway Cfg Byte Offset	60h:63h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	ROMDATA	R/W	0	Contains 32-bits of data that is either written to the ROM or has been read from the ROM. When a parallel ROM byte access is performed, only bits [7:0] are used for data (regardless of the byte alignment of the address). Read data is valid after the SG2010 has cleared the corresponding Start/Busy bit.

4.8.6 Enhanced Bridge Addressing Capabilities

These registers are used by the Bridge to enable additional addressing capabilities on the secondary bus when the SG2010 is a leaf; that is, the PCI bus is the secondary bus. These registers are not used when the Bridge is a root bridge (PCI is primary).

The secondary base/limit is used to selectively block upstream forwarding. This feature can be configured such that inversely decoded transactions inside the address range are blocked (ignored) by the SG2010, while they are forwarded outside the address range. It can also be configured so that inversely decoded transactions are blocked (ignored) by the SG2010 outside the address range, and only forwarded upstream if they fall within the address range.

The Secondary IDSEL mask is used to selectively hide secondary bus devices from the host.

4.8.6.1 Secondary Memory Base

Gateway Cfg Byte Offset	64h:65h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
0	SMEM64	R	1	Reads as 1 to indicate that the SG2010 supports a 64-bit secondary memory address range.
3:1	RES	R	0	Reserved.
15:4	SMBASE	R/W	0	Defines address bits [31:20] of the low end of the secondary memory address range. Bits [19:0] are assumed to be 0, giving a minimum size and alignment granularity of 1MB.

4.8.6.2 Secondary Memory Limit

Gateway Cfg Byte Offset	66h:67h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
0	SMEM64	R	1	Reads as 1 to indicate that the SG2010 supports a 64-bit secondary memory address range.
3:1	RES	R	0	Reserved.
15:4	SMLIMIT	R/W	0	Defines address bits [31:20] of the high end of the secondary memory address range. Bits [19:0] of the address are assumed to be 1s, giving a minimum size and alignment granularity of 1MB.

4.8.6.3 Secondary Memory Base Upper 32 Bits

Gateway Cfg Byte Offset	68h:6Bh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	SMB64	R/W	0	Defines address bits [63:32] of the low end of the secondary memory address range.

4.8.6.4 Secondary Memory Limit Upper 32 Bits

Gateway Cfg Byte Offset	6Ch:6Fh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	PFML64	R/W	0	Defines address bits [63:32] of the high end of the secondary memory address range.

4.8.6.5 Secondary IDSEL Mask

Gateway Cfg Byte Offset	70h:71h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
15:0	IDMASK	R/W	0	Defines an IDSEL mask for Type1 to Type0 configuration transactions forwarded by the Bridge to the secondary PCI bus. The IDSEL signals are generated from the upper 16 bits of the AD bus. If a corresponding bit in the 16-bit IDSEL mask is set, the SG2010 forces that address bit to 0 whenever it initiates a Type0 configuration transaction on the secondary bus. If the IDSEL mask bit is not set, then the SG2010 drives whatever bit is decoded from the device number in the received Type1 frame.

4.8.7 Address Setup Registers

The following registers set the type and size of their respective BARs. The amount of space requested by a BAR is programmed with a bit pattern in the BASESIZE address field of the corresponding setup register. A 1 in a bit location sets the corresponding BAR bit to R/W, and 0 in a bit location sets the corresponding BAR bit to RO. A valid pattern in this field consists of zero or more contiguous 1s in the high bits, followed by zero or more contiguous 0s in the low bits. This pattern must occupy the entire BASE-SIZE field.

Note: *An invalid pattern can cause unpredictable results.*

4.8.7.1 PCI BAR2 Setup and PCI BAR4 Setup

These two setup registers configures their respective BARs to be 32-bit BARs or the lower 32-bits of a 64-bit bar.

PCI Bar 2 Setup Gateway Cfg Byte Offset	80h:83h
PCI Bar 4 Setup Gateway Cfg Byte Offset	88h:8Bh
Size	4 bytes each

Bit	Name	Access	Reset Value	Description
0	RES	R	0	Reserved.
2:1	TYPE	R/W	0	Indicates the size of the corresponding BAR and its mapping requirements. Allowable values are 00b (32-bit, map anywhere) 10b (64-bit) *Any other values will cause unpredictable results.*
3	PF	R/W	0	When 1, indicates that the address range of the corresponding BAR is prefetchable.
18:4	RES	R	0	Reserved
30:19	BASESIZE	R/W	0	Programs the size of the address range (power-of-2 granularity) for the BAR corresponding to this register.
31	BARENA or BASESIZE	R/W	0	If this BAR is set up to be 32-bit and this bit is 1, the BAR is enabled and uses the rest of the setup register data to set BAR parameters. When this bit is a 0, the BAR is disabled (all BAR bits read only as 0 and no addresses are decoded against it). If this BAR is set up to be 64-bits, this bit is another bit in the base size field.

4.8.7.2 PCI BAR3 Setup and PCI BAR5 Setup

These two setup registers configures their respective BARs to be 32-bit BARs or the upper 32-bits of a 64-bit bar.

PCI Bar 3 Setup Gateway Cfg Byte Offset	84h:87h
PCI Bar 5 Setup Gateway Cfg Byte Offset	8Ch:8Fh
Size	4 bytes each

Setup register definition if configuring a 32-bit BAR:

Bit	Name	Access	Reset Value	Description
0	UNDEF	R/W	0	These bits can be written and read, but are ignored by the SG2010 when configuring a 32-bit BAR.
2:1	TYPE	R/W	0	Indicates the size of the corresponding BAR and its mapping requirements. Allowable values are 00b (32-bit, map anywhere) 10b (64-bit) *Any other values will cause unpredictable results.*
3	PF	R/W	0	When 1, indicates that the address range of the corresponding BAR is prefetchable.

Bit	Name	Access	Reset Value	Description
18:4	UNDEF	R/W	0	These bits can be written and read but do not affect the size of a 32-bit BAR. The SG2010 ignores these bits when determining the size of a 32-bit BAR. The minimum size of a 32-bit BAR is 512KB.
30:19	BASESIZE	R/W	0	Programs the size of the address range (power-of-2 granularity) for the BAR corresponding to this register.
31	BARENA	R/W	0	If this BAR is set up to be 32-bit and this bit is a 1, the BAR is enabled and uses the rest of the setup register data to set BAR parameters. When this bit is a 0, the BAR is disabled (all BAR bits read only as 0 and no addresses are decoded against it).

Setup register definition if configuring a 64-bit BAR:

Bit	Name	Access	Reset Value	Description
19:0	BASESIZE	R/W	0	Programs the size of the address range (power-of-2 granularity) for the 64-bit BAR corresponding to this register.
30:20	UNDEF	R/W	0	These bits can be written and read, but do not affect the size of a 64-bit BAR. The SG2010 ignores these bits when determining the size of a 64-bit BAR. The size of a 64-bit BAR is limited to 1PB (2^{50} bytes) or less.
31	BARENA	R/W	0	When a 1, enables the BAR and uses the rest of the setup register data to set BAR parameters. When a 0, the BAR is disabled (all BAR bits read only as 0 and no addresses are decoded against it). If this is the upper 32 bits of a 64-bit BAR, acts as an enable for the entire BAR.

4.8.7.3 Expansion ROM Setup

Gateway Cfg Byte Offset	90h:93h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
11:0	RES	R	0	Reserved
23:12	BASESIZE	R/W	0	Programs the size of the address range (power-of-2 granularity) for the BAR corresponding to this register.
24	BARENA	R/W	0	When a 1, enables the BAR and uses the rest of the setup register data to set BAR parameters. When a 0, the BAR is disabled (all BAR bits read only as 0 and no addresses are decoded against it).
31:25	RES	R	0	Reserved.

4.8.8 Device-specific Control Registers

4.8.8.1 Chip Control

Gateway Cfg Byte Offset	94h:97h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
0	CFGLOCK	R/W0 (S/W) W (SROM)	H/W	Lockout bit. When the SG2010 is a leaf and this bit is set to a 1, the SG2010 returns a completion frame with the lockout status set in response to all incoming address-routed frames targeted to SG2010 registers. When this bit is 0, the SG2010 accepts all incoming frames. This bit can be set either by pulling the LOCKOUT strapping pin high at reset or by an SROM preload write. Software cannot write this bit to 1. This bit can be cleared either by pulling the LOCKOUT strapping pin low during reset, by an SROM preload write, or by software writing 0. A serial preload write overrides the value set by the strapping pin.
2:1	SECMODE	R/W	00b	Secondary Base/Limit Mode. Enables or disables the secondary base and limit and determines how it is used. 00b: Secondary Base and Limit window is disabled; the SG2010 responds normally to all inversely decoded upstream transactions. 01b: The SG2010 does not respond to inversely decoded upstream transactions *outside* of the Secondary Base/Limit window. 10b: The SG2010 does not respond to inversely decoded upstream transactions *inside* of the Secondary Base/Limit window. 11b: Reserved encoding.

Bit	Name	Access	Reset Value	Description
3	PFDIS	R/W	0b	Upstream prefetch disable for Bridge function. When: 0 Upstream transactions using the Memory Read command are treated as prefetchable and attempt to fetch the amount of data dictated by the Memory Read Prefetch bits. 1 Upstream PCI transactions using the Memory Read command are treated as non-prefetchable; the SG2010 requests only a single Dword for these transactions. Only meaningful for a leaf; the root always requests the amount indicated in the read request frame.
5:4	MRPF	R/W	00B	Memory Read Prefetch amount for speculative memory reads. Sets the amount of data to be requested for Memory Read transactions going from PCI to StarFabric. Encodings are: 00b: 1 cache line 01b: 2 cache lines 10b: 4 cache lines 11b: 1 Dword
7:6	MRLPF	R/W	00b	Memory Read Line Prefetch amount for speculative memory reads. Sets the amount of data to be requested for Memory Read Line transactions going from PCI to StarFabric. Encodings are: 00b: 1 cache line 01b: 2 cache lines 10b: 4 cache lines 11b: 8 cache lines
9:8	MRMPF	R/W	00b	Memory Read Multiple Prefetch amount for speculative memory reads. Sets the amount of data to be requested for Memory Read Multiple transactions going from PCI to StarFabric. Encodings are: 00b: 2 cache line 01b: 4 cache lines 10b: 8 cache lines 11b: 16 cache lines
10	MTSEL	R/W	0b	Master Time-out Select. Selects the expiration time of the Master Time-out Discard timer for the Gateway function. When: 0 The timer is set to expire after 2^{15} PCI clock cycles. 1 The timer is set to expire after 2^{10} PCI clock cycles.

Bit	Name	Access	Reset Value	Description
11	RETENA	R/W	0b	Read Data Retention Enable. When: 0 The SG2010 discards speculative prefetched read data after the transaction is master terminated or disconnected on the PCI bus. 1 Enables the SG2010 to retain speculative prefetched read data when a PCI read transaction is master terminated or disconnected.
12	COMBENA	R/W	0b	Write Combine Enable for transactions initiated on the PCI bus. When: 0 The SG2010 does not perform write combining and each write frame is translated into a separate transaction. 1 The SG2010 is enabled to combine memory write transactions initiated on the PCI bus, when combining rules allow.
13	SGCTENA	R/W	0b	Configuration SGT Enable. When: 0 Accessing the Configuration SGT Data register has no effect – writes are discarded and read data is undefined. 1 Enables the Configuration SGT function. That is, when the Configuration SGT Data register is read or written from the StarFabric interface, the SG2010 initiates a configuration read or write on the PCI bus.
14	SGIOTENA	R/W	0b	I/O SGT Enable. When: 0 Accessing the I/O SGT Data register has no effect – writes are discarded and read data is undefined. 1 Enables the I/O SGT function. That is, when the I/O SGT Data register is read or written from the StarFabric interface, the SG2010 initiates an I/O read or write on the PCI bus.
15	GWMASK	R/W	0b	Gateway configuration mask. When: 0 Configuration read and write frames to the SG2010 are handled normally. 1 And the SG2010 is a leaf with Bridge function enabled, configuration read and write frames to the Gateway from the StarFabric interface result in a failure type of Address Routing Failure. That is, the Gateway device is hidden from the host.
20:16	GWDEVNUM	R/W	1Fh	Identifies the device number of the Gateway when the SG2010 is a leaf and the Bridge function is enabled. Used when decoding Gateway configuration transactions as a leaf. All values from 0 to 31 are supported. Defaults to device number 31.

Bit	Name	Access	Reset Value	Description
21	HIDEBARS	R/W	0b	BAR2 – BAR5 Visibility Mode. This bit is only meaningful when the SG2010 is a leaf. If the SG2010 is a root the state of this bit is ignored. When 0 All Gateway BARs are visible both to the host and on the secondary PCI bus. 1 BARs 2, 3, 4 and 5 are hidden from the host; that is, configuration transactions from the link accessing BAR2, BAR3, BAR4, and BAR5 receive 0s on a read and discard data on a write, but are accessible to Type0 configuration transactions from the secondary PCI bus. All other Gateway configuration registers are visible to both sides and the SG2010 responds normally.
22	OTQMODE	R/W	0	Specifies the CoS reservation mode of both the PCI Delayed Transaction Buffer and the Outstanding Transaction Buffer. When: 0 The SG2010 uses a CoS reservation mechanism that guarantees every supported CoS at least one entry in each of the two buffers. 1 The SG2010 does not use a CoS reservation mechanism – any queue entry in these buffers can be used by a transaction of any CoS.
23	TTMODE	R/W	0	Transaction termination mode. Determines when the SG2010 terminates a PCI transaction in order to initiate a different transaction. When: 0 The SG2010 does not terminate a PCI transaction in order to initiate another PCI transaction – only the master latency timer puts a limit on the duration of the transaction. This mode optimizes throughput. 1 The SG2010 terminates a transaction on an aligned 32 dword boundary after 32 PCI clock cycles have passed since the first TRDY_L assertion, if it has an outgoing transaction pending. This mode optimizes latency.
25:24	ENUMTMR	R/W	10b	Fabric Enumeration Timer Control. Set during the first byte of serial preload and selects one of four initial values for the fabric enumeration timer. 00b: ≈2.5ms 01b: ≈5.0ms 10b: ≈10ms 11b: ≈20ms

Bit	Name	Access	Reset Value	Description
26	SKIPINSENA	R/W	H/W	Skip Insertion Enable. When: 0 The hot swap controller must transition through the Insertion state in order to get to the Installed state on power-up. For remote ENUM# assertion, the Remote INS Ready bit must be set to transition to the H/W Connected state. 1 The hot swap controller skips the Insertion state on power-up. For remote ENUM# assertion, transition to the H/W Connected state is not gated by Remote INS Ready. A strapping pin shared with PR_AD[2], sampled on the deasserting edge of LRST_L or RST_L, sets the reset value of this bit. This bit is only reset by LRST_L or RST_L assertion and cannot be reset by other reset mechanisms.
27	RINSRDY	R/W	0b	Remote INS Ready. Only meaningful in the H/W Connected state of the hot swap state machine. When: 0 Prevents the state machine from transitioning out of this state when a remote ENUM_L is specified and the Skip Insertion Enable is not set. 1 Enables the transition from the H/W Connected state when serial preload is complete. This bit is only reset by LRST_L or RST_L assertion and cannot be reset by other reset mechanisms.
28	PCITR	R/W	0	PCI Target Response Mode. In conjunction with the PCI Master Abort Mode bit in the Bridge Control register, controls SG2010's response on the PCI bus to a variety of failures. 0 When this bit and the Master Abort Mode bit are both 0, a benign response is returned; that is, write data is discarded and reads return FFFFFFFFh. 1 When this bit or the Master Abort Mode bit is a 1, an error response is returned – a target abort is returned for delayed transactions and a system error event is signaled for writes without acknowledges. Table 3–8 lists the failures where this bit is used.

Bit	Name	Access	Reset Value	Description
29	MRSTM	R/W	0	Maskable Reset Mode. When 0, if the SG2010 receives a maskable reset comma, it propagates the reset and resets the chip if the Reset Disable bit in the Port State Table is set for that link. When 1, if the SG2010 receives a maskable reset comma, it clears the Traffic Enable bit and resets the FID if its Fabric ID has been enumerated, or sets the Traffic Enable bit if the Fabric ID has not been enumerated. The chip does not reset.
30	VGA16S	R/W	0	VGA 16-bit Decode Supported. When 0, bit [4] in the Bridge Control register in Bridge configuration space and in the Port Map Tables is read only as 0. VGA 16-bit Decode cannot be enabled. When 1, bit [4] in the Bridge Control register in Bridge configuration space and in the Port Map Tables is read/write. VGA 16-bit Decode may be enabled by setting that bit.
31	RES	R	0	Reserved.

4.8.8.2 Chip Status

Gateway Cfg Byte Offset	98h:99h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
0	ROOT	R	H/W	Reflects the value of the Root pin. When: 0 The SG2010 is a leaf. 1 The SG2010 is a root.
1	PFN0	R	H/W	Reflects the value of the PFN0 strapping pin as it was captured on the deasserting edge of RST_L or LRST_L. The PFN0 pin is shared with PR_AD[4] and is sampled at the deassertion of reset. The SG2010 uses this state to set the value of PFN[0] when it performs fabric enumeration as a root.
2	BRENA	R	H/W	Reflects the value of the Bridge Enable pin. When: 0 The Bridge is disabled and only the Gateway is visible. 1 The Bridge function is enabled in multi-function mode if a root bridge and secondary subordinate mode if a leaf bridge.

Bit	Name	Access	Reset Value	Description
3	LOCK	R	H/W	Reflects the value of the Lockout strapping pin as it was captured on the deasserting edge of RST_L or LRST_L. The Lockout pin is shared with PR_AD[5] and is sampled at the deassertion of reset. When: 0 The Configuration Lockout bit is initialized to 0. 1 The Configuration Lockout bit is initialized to 1.
4	PCIARB	R	H/W	Reflects the value of the Arbiter Enable strapping pin as it was captured on the deasserting edge of RST_L or LRST_L. The Arbiter Enable pin is shared with PR_AD[6] and is sampled at the deassertion of reset. When: 0 The PCI arbiter is disabled and an external arbiter must be used. 1 SG2010's PCI arbiter is enabled.
5	CFENA	R	H/W	Reflects the value of the PCI Central Function Enable strapping pin as it is captured during RST_L or LRST_L. The Central Function Enable pin is shared with PR_AD[7] and is sampled during reset. When: 0 The SG2010 does not perform central functions on the PCI bus during reset. 1 The SG2010 does perform central functions on the PCI bus during reset. Central functions include asserting REQ64_L during reset, and driving zeros on AD[31:0], CBE_L[3:0] and PAR. Central functions are synchronous to RSTO_L.
6	P64EN	R	H/W	When: 0 The SG2010 PCI 64-bit extension interface is not enabled – all PCI transactions are 32 bits. 1 The SG2010 PCI 64-bit extension interface is enabled to perform 64-bit transactions.
7	P66EN	R	H/W	Reflects the state of the M66ENA pin. When: 0 The SG2010 PCI interface is operating at or below 33MHz. 1 The SG2010 PCI interface is operating between 33MHz and 66MHz.
8	LSTAT	R	H/W	Reflects the debounced state of signal LSTAT.
9	BDSEL	R	H/W	Reflects the debounced state of signal BDSEL_L.
12:10	RES	R	0	Reserved.
13	CSGTBUSY	R	0	Configuration SGT busy. Set by the SG2010 when the SGT Configuration Address register is written and cleared when the SGT configuration access is complete. Does not affect access to the SGT function.

Bit	Name	Access	Reset Value	Description
14	ISGTBUSY	R	0	I/O SGT busy. Set by the SG2010 when the SGT I/O Address register is written and cleared when the SGT I/O access is complete. Does not affect access to the SGT function.
15	RES	R	0	Reserved

4.8.8.3 Arbiter Control

Gateway Cfg Byte Offset	9Ah:9Bh
Size	2 bytes

Bit	Name	Access	Reset Value	Description
1:0	APRI	R/W	01b	SG2010 arbiter priority. Sets the priority of the SG2010 as a bus master in SG2010's PCI arbiter. Possible encodings are: 00b: Low priority group 01b: One slot, high priority group 10b: Two slots, high priority group 11b: Reserved
3:2	BM0PRI	R/W	00b	Bus Master 0 priority. Sets the priority of the bus master using REQ_L[0] and GNT_L[0]. Possible encodings are: 00b: Low priority group 01b: One slot, high priority group 10b: Two slots, high priority group 11b: Reserved
5:4	BM1PRI	R/W	00b	Bus Master 1 priority. Sets the priority of the bus master using REQ_L[1] and GNT_L[1]. Possible encodings are: 00b: Low priority group 01b: One slot, high priority group 10b: Two slots, high priority group 11b: Reserved
6	BM2PRI	R/W	0	Bus Master 2 priority. Sets the priority of the bus master using REQ_L[2] and GNT_L[2]. When: 0 Bus master occupies one slot in the low priority group. 1 Bus master occupies one slot in the high priority group.
7	BM3PRI	R/W	0	Bus Master 3 priority. Sets the priority of the bus master using REQ_L[3] and GNT_L[3]. When: 0 Bus master occupies one slot in the low priority group. 1 Bus master occupies one slot in the high priority group.
8	BM4PRI	R/W	0	Bus Master 4 priority. Sets the priority of the bus master using REQ_L[4] and GNT_L[4]. When: 0 Bus master occupies one slot in the low priority group. 1 Bus master occupies one slot in the high priority group.

Bit	Name	Access	Reset Value	Description
9	BM5PRI	R/W	0	Bus Master 5 priority. Sets the priority of the bus master using REQ_L[5] and GNT_L[5]. When: 0 Bus master occupies one slot in the low priority group. 1 Bus master occupies one slot in the high priority group.
10	BM6PRI	R/W	0	Bus Master 6 priority. Sets the priority of the bus master using REQ_L[6] and GNT_L[6]. When: 0 Bus master occupies one slot in the low priority group. 1 Bus master occupies one slot in the high priority group.
11	BM7PRI	R/W	0	Bus Master 7 priority. Sets the priority of the bus master using REQ_L[7] and GNT_L[7]. When: 0 Bus master occupies one slot in the low priority group. 1 Bus master occupies one slot in the high priority group.
12	BM8PRI	R/W	0	Bus Master 8 priority. Sets the priority of the bus master using REQ_L[8] and GNT_L[8]. When: 0 Bus master occupies one slot in the low priority group. 1 Bus master occupies one slot in the high priority group.
13	RES	R	0	Reserved
14	PARK	R/W	0	Bus Parking Control. Selects the bus parking mode for the PCI bus AD[31:0], CBE_L[3:0] and PAR when the arbiter is enabled. When: 0 The bus is always parked at the SG2010. 1 The bus is parked at the last master to use the bus.
15	PINMODE	R/W	0	Arbiter pin mode select. When 0, signal pins REQ_L[8:7] and GNT_L[8:7] are assigned to the arbiter. When a 1, the pins are assigned to the GPIO function as follows: REQ_L[7]:GPIO[4] GNT_L[7]:GPIO[5] REQ_L[8]:GPIO[6] GNT_L[8]:GPIO[7]

4.8.9 General Purpose I/O (GPIO) Registers

4.8.9.1 GPIO Data Set

Gateway Cfg Byte Offset	A0h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	SGPDAT	R/W1TS	0	GPIO Data Set operation. When read, returns the current value of the GPIO signal pins. When written with 1, sets the corresponding bit in the GPIO Data register. If the same bit in the GPIO Direction register is 1, then the corresponding GPIO signal pin is driven high and stays high until cleared or until the direction is changed. Writing this register with 0 has no effect.

4.8.9.2 GPIO Direction Set

Gateway Cfg Byte Offset	A1h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	SGPDIR	R/W1TS	0	GPIO Direction Set operation. When read, returns the current direction of each GPIO signal pin, where 0 indicates that the pin is input-only and 1 indicated that the pin is bidirectional. When any bit is written with 1, the corresponding GPIO pin is configured as a bidirectional pin and the current value of the same bit of the GPIO Data register is driven onto the pin. If GPIO[7:4] are assigned to the arbiter, the direction of the pins cannot be set to bidirectional and the corresponding bits read as 0. Writing this register with 0 has no effect.

4.8.9.3 GPIO Data Clear

Gateway Cfg Byte Offset	A2h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	CGPDAT	R/W1TC	0	GPIO Data Clear operation. When read, returns the current value of the GPIO Data register. When written with 1, clears the corresponding bit in the GPIO Data register. If the same bit in the GPIO Direction register is 1, then the corresponding GPIO signal pin is driven low and stays low until set or until the direction is changed. Writing this register with 0 has no effect.

4.8.9.4 GPIO Direction Clear

Gateway Cfg Byte Offset	A3h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	CGPDIR	R/W1TC	0	GPIO Direction Clear operation. When read, returns the current direction of each GPIO signal pin, where 0 indicates that the pin is input-only and 1 indicated that the pin is bidirectional. When any bit is written with 1, the corresponding GPIO pin is configured as an input only pin. Reading the GPIO Data register returns the current level of all the GPIO pins. Writing this register with 0 has no effect.

4.8.10 Gateway Power Management Registers

4.8.10.1 Power Management ECP ID

Gateway Cfg Byte Offset	AC
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	PMECPID	R	01h	Extended Capabilities ID for the power management function. Must read as 01h.

4.8.10.2 Power Management Next Pointer

Gateway Cfg Byte Offset	ADh
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	PMNPTR	R	BCh	Contains the configuration offset of the next ECP function, which is MSI at offset BCh.

4.8.10.3 Power Management Capabilities

Gateway Cfg Byte Offset	AEh:AFh
Size	2 bytes

Bit	Name	Access	Reset Value	Description
2:0	PMVER	R	2h	Reads as 010b to indicate that this device complies with version 1.1 of the PCI Power Management specification.
3	PMECLK	R	0	PCI clock required for PME#. This function does not support PME_L and this bits reads as 0.
4	RES	R	0	Reserved.
5	DSI	R	0	Device Specific Initialization required. Reads as 0 to indicate that device specific initialization is not required following a transition to D0.
8:6	AUXCURR	R	0	3.3V auxiliary current requirements. This function does not support PME_L and these bits read as 0.
9	D1_SUPP	R	0	D1 Support. Reads as 0 to indicate that this function does not support the D1 power state.
10	D2_SUPP	R	0	D2 Support. Reads as 0 to indicate that this function does not support the D2 power state.
15:11	PME_SUPP	R	0	PME# Support. Reads as 0 to indicate that PME_L assertion is not supported in any power state.

4.8.10.4 Power Management Control and Status

Gateway Cfg Byte Offset	B0h:B1h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
1:0	PWRST	R/W	0	Indicates the current power state of this function and controls transitions to a new power state. Only D0 (00b) and D3 (11b) are supported, and only writes to these states are allowed. The SG2010 will remain in the current power state if a write to either D1 or D2 is attempted.
7:2	RES	R	0	Reserved.
8	PME_EN	R	0	PME# Enable. Reads as 0 because PME_L is not supported by this function.
12:9	DATA_SEL	R	0	Data Select Index. Reads as 0 because the Data register is not supported.
14:13	SCALE	R	0	Data Scale. Reads as 0 because the Data register is not supported.
15	PME_ST	R	0	PME# Status. Reads as 0 because this function does not support PME_L.

4.8.10.5 Power Management Data

Gateway Cfg Byte Offset	B3h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	PMDAT	R	0	Data register. Not supported, reads as 0.

4.8.11 VPD ECP Registers

These registers are a dual-mapped copy of the VPD registers in the Bridge configuration registers. There is only a single VPD function.

4.8.11.1 VPD ECP ID

Gateway Cfg Byte Offset	B4h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	VPDECPID	R	3h	Extended Capabilities ID for the VPD function. Must read as 3h.

4.8.11.2 VPD Next Pointer

Gateway Cfg Byte Offset	B5h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	VPDNXTPTR	R	0	Indicates the next ECP function in the linked list. Because VPD is the last function, reads as 0.

4.8.11.3 VPD Address

Gateway Cfg Byte Offset	B6h:B7h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
6:0	VPDADDR	R/W	0	Dword aligned byte-address offset of the 128-byte VPD address region to be accessed (bits [1:0] are ignored and assumed to be 00b).
14:7	RES	R	0	Reserved
15	VPDFLAG	R/W	0	VPD operation/status bit. When written with: 0 The SG2010 reads four bytes of data from the VPD address location written to bits [6:0]. When the read is complete and data is available, SG2010 sets this bit to 1. 1 The SG2010 writes four bytes of data contained in the VPD data register to the VPD address location written to bits [6:0]. When the write is complete, SG2010 sets this bit to 0.

4.8.11.4 VPD Data

Gateway Cfg Byte Offset	B8h:BBh
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	VPDDATA	R/W	0	Contains four bytes of VPD data. On a VPD read, SG2010 places read data in this register. Before a VPD write operation, software must write the data to be stored to this register.

4.8.12 Message Signaling Interrupt (MSI) Registers

4.8.12.1 MSI ECP ID

Gateway Cfg Byte Offset	BCh
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	MSIECPID	R	05h	Extended Capabilities ID for the MSI function. Must read as 05h.

4.8.12.2 MSI Next Pointer

Gateway Cfg Byte Offset	BDh
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	MSINPTR	R	B4[*] D8[†]	Contains the configuration offset of the next ECP function. If the Bridge function is enabled, the next function is VPD and this register returns B4. If the Bridge function is disabled, the next function is hot swap and this register returns D8.

* When Bridge function is enabled.
† When Bridge function is disabled.

4.8.12.3 MSI Message Control

Gateway Cfg Byte Offset	BEh:BFh
Size	2 bytes

Bit	Name	Access	Reset Value	Description
0	MSIENA	R/W	0	MSI Enable. When: 0 MSI for this function is disabled and the INTx_L pin must be used for interrupts. 1 This function is enabled to use MSI and cannot use an INTx_L pin to assert an interrupt. The SG2010 supports individual INTx_L pins enabled for MSI through the Event Dispatch Control CSR. By default, only local INTA_L is enabled to use MSI, if this MSI Enable bit is set.
3:1	MULTMSG	R	010b	Indicates the number of interrupt messages that this function is requesting. The SG2010 requests four messages, which is encoding 010b.

Bit	Name	Access	Reset Value	Description
6:4	MULTENA	R/W	0	Multiple Message Enable. Software writes this field with the number of interrupt messages allocated. The possible encodings are: 000b: 1 message 001b: 2 messages 010b: 4 messages 011b: 8 messages 100b: 16 messages 101b: 32 messages 110b: Reserved 111b: Reserved
7	MSI64	R	1	MSI 64-bit capable. Reads as 0 to indicate that this MSI function is capable of generating 64-bit addresses.
15:8	RES	R	0	Reserved.

4.8.12.4 MSI Message Address

Gateway Cfg Byte Offset	C0h:C3h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
1:0	RES	R	0	Reserved
31:2	MSIADDR	R/W	0	Specifies the low 32 bits of the Dword-aligned address to be used by the SG2010 for MSI transactions.

4.8.12.5 MSI Message Address Upper 32 Bits

Gateway Cfg Byte Offset	C4h:C7h
Size	4 bytes

Bit	Name	Access	Reset Value	Description
31:0	MSIADDR64	R/W	0	Specifies the upper 32 bits of the Dword-aligned address to be used by the SG2010 for MSI transactions.

4.8.12.6 MSI Message Data

Gateway Cfg Byte Offset	C8h:C9h
Size	2 bytes

Bit	Name	Access	Reset Value	Description
15:0	MSIDAT	R/W	0	Contains the system-specified MSI data used for MSI transactions. The SG2010 may modify the low significant bits corresponding to the number of messages it has. 1 or 2 messages enabled: The SG2010 modifies no bits. 4 messages: The SG2010 modifies bits [1:0] as follows - 00b: INTA_L assertion 01b: INTB_L assertion 10b: INTC_L assertion 11b: INTD_L assertion

4.8.13 Compact PCI Hot Swap Registers

These registers are visible only when the Bridge function is disabled. If the Bridge function is enabled, it is assumed the hot swap function is accessed through the Bridge. If the Bridge function is enabled, Dword D8h is reserved and returns 0 when read.

Only the assertion of LRST_L or RST_L resets the hot swap register bits; they cannot be reset by other mechanisms.

4.8.13.1 Hot Swap ECP ID

GW Cfg Byte Offset	D8h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	HSECPID	R	6h	Extended Capabilities ID for the hot swap function. Must read as 06h.

4.8.13.2 Hot Swap Next Pointer

GW Cfg Byte Offset	D9h
Size	1 byte

Bit	Name	Access	Reset Value	Description
7:0	HSNPTR	R	B4	Contains the configuration offset of the next ECP function, which is VPD at offset B4h.

4.8.13.3 Hot Swap Control

GW Cfg Byte Offset	DAh
Size	1 byte

Bit	Name	Access	Reset Value	Description
0	DHE	R/W	0	Device Hiding Enable. Implemented when PI = 1. When: 0 Device hiding is disabled. 1 Device hiding is enabled. When PI = 0 this bit is read-only and returns 0.
1	EIM	R/W	0	ENUM_L interrupt mask. When: 0 Enables the assertion of ENUM_L whenever INS or EXT are set. 1 Disables the assertion of ENUM_L by this hot swap function.
2	PIE	R	H/W	Pending Insertion/Extraction. Implemented when PI=1. Reads as: 0 When the SG2010 is in the installed state. 1 When an insertion or extraction is in progress.
3	LOO	R/W	0	LED on/off. When the LED is under software control and LOO is: 0 The HS_LED signal is deasserted and the LED is off. 1 The HS_LED signal is asserted and the LED is illuminated.
5:4	PI	R	1	Programming Interface. Indicates the Compact PCI Hot Swap programming interface revision. By default 1 this bit is, but can be cleared to 0 through serial preload. When: 0 Only the EIM, LOO, EXT and INS bits are implemented; the remaining bits in this register read only as 0. Device hiding and software-initiated extractions are not implemented. 1 All bits are implemented, and device hiding and software-initiated extractions are supported.
7	INS	R/W1TC	0	Board inserted status bit. Set by the SG2010 when the device is ready for configuration, after ejector handle has been closed (LSTAT = 0), reset has been deasserted, and any local initialization has been completed. Software clears this bit by writing a 1 to this location. Writing a 0 has no effect.

STARGEN 5

Signal Pin Descriptions

Signal Type	Description
I	Input only.
O	Output only.
TS	Tristate bidirectional.
STS	Sustained tristate bidirectional.
OD	Open drain output only.
BOD	Bidirectional open drain.
LO	LVDS output
LI	LVDS input

5.1 PCI Interface Signal Pins

Signal Name	Width	Type	Description
AD[63:0]	64	TS	PCI multiplexed address/data bus. The upper 32 bits of this bus are enabled when REQ64_L is sampled low on RST_L or LRST_L deassertion, or when L64EN_L is detected low. When disabled, the SG2010 drives the upper 32 bits to a valid logic level and only the lower 32 bits are used for PCI transactions. AD[63:32] are not bus parked and must be pulled up through external resistors.
CBE_L[7:0]	8	TS	PCI multiplexed command/byte enable bus. The upper 4 bits of this bus are enabled when REQ64_L is sampled low on RST_L or LRST_L deassertion, or when L64EN_L is detected low. When disabled, the SG2010 drives the upper 4 bits to a valid logic level and only the lower 4 bits are used for PCI transactions. CBE_L[7:4] are not bus parked and must be pulled up through external resistors.
PAR	1	TS	PCI parity pin. This pin is driven by the device driving the data one cycle after data is driven and creates even parity covering the AD[31:0] and CBE_L[3:0] pins.

PCI Interface Signal Pins

Signal Name	Width	Type	Description
PAR64	1	TS	PCI 64-bit extension parity pin. This pin is driven by the device driving the data one cycle after data is driven and creates even parity covering the AD[63:32] and CBE_L[7:4] pins. This signal is enabled when REQ64_L is sampled low on RST_L or LRST_L deassertion, or when L64EN_L is detected low. When disabled, the SG2010 drives this signal to a valid logic level. PAR64 is not bus parked and must be pulled up through an external resistor.
REQ64_L	1	STS	PCI 64-bit transaction request. Driven low by the initiator of a PCI transaction when the initiator wishes to conduct the transaction using the 64-bit extension. REQ64_L is driven with the same timing as FRAME_L. When sampled low on the deassertion of RST_L or LRST_L, enables the 64-bit extension signals. The SG2010 drives this signal low during RSTO_L assertion if the central function is enabled.
FRAME_L	1	STS	PCI transaction frame. Driven by the initiator of a PCI transaction to indicate the start and duration of a transaction. The initiator drives the signal low during at the beginning of the address phase and deasserts the transaction at the beginning of the last data phase.
IRDY_L	1	STS	PCI initiator ready. Driven by the initiator of a PCI transaction indicating that the initiator is either driving valid write data or ready to accept read data. A data transfer occurs when both IRDY_L and TRDY_L are asserted.
IDSEL	1	TS	PCI configuration device select. Sampled by the SG2010 to determine whether it is the target of a PCI Type0 configuration read or write transaction.
DEVSEL_L	1	STS	PCI target device select. Driven by a PCI device indicating that it has determined that it is the target of a PCI transaction after successfully decoding the transaction address. The target deasserts this signal at the end of the last data phase.
ACK64_L	1	STS	PCI 64-bit transaction acknowledge. Driven low by the target of a transaction in response to the initiator's assertion of REQ64_L to indicate that it will perform the transaction as a 64-bit transaction. Driven with the same timing as DEVSEL_L.
TRDY_L	1	STS	PCI target ready. Driven by the target of a PCI transaction indicating that the target is either driving valid read data or ready to accept write data. A data transfer occurs when both IRDY_L and TRDY_L are asserted.
STOP_L	1	STS	PCI target transaction termination. Driven by the target of a PCI transaction indicating that it is forcing the current data phase to be the last data phase. May be driven with TRDY_L to perform a target disconnect with data or without TRDY_L to terminate the transaction without transferring data during that data phase.
PERR_L	1	STS	PCI parity error detected. Asserted by the device receiving data when it detects a parity error on the received data and its Parity Error Response bit is set. Asserted two cycles after the data with error is driven and one cycle after parity is driven.
SERR_L	1	BOD	PCI system error. Asserted by a PCI device when it encounters a severe error. The SG2010 can be enabled to sample this signal and forward it through the event dispatcher, and can be enabled to assert this signal when it receives an event directed to the SERR_L EMU address.

Signal Name	Width	Type	Description
INTA_L	1	BOD	PCI device interrupt signal A. The SG2010 can be enabled to sample this signal and forward it through the event dispatcher, and can be enabled to assert this signal when it receives an event directed to the local INTA_L or remote INTA_L EMU address.
INTB_L	1	BOD	PCI device interrupt signal B. The SG2010 can be enabled to sample this signal and forward it through the event dispatcher, and can be enabled to assert this signal when it receives an event directed to the INTB_L EMU address.
INTC_L	1	BOD	PCI device interrupt signal C. The SG2010 can be enabled to sample this signal and forward it through the event dispatcher, and can be enabled to assert this signal when it receives an event directed to the INTC_L EMU address.
INTD_L	1	BOD	PCI device interrupt signal D. The SG2010 can be enabled to sample this signal and forward it through the event dispatcher, and can be enabled to assert this signal when it receives an event directed to the INTD_L EMU address.
PME_L	1	BOD	PCI power management interrupt. The SG2010 can be enabled to sample this signal and forward it through the event dispatcher, and can be enabled to assert this signal when it receives an event directed to the PME_L EMU address.
CLK	1	I	PCI clock input. Controls SG2010's PCI interface logic, register access logic, and ROM interface logic. When M66EN is low, the SG2010 supports a functional CLK ranging from 25MHz to 33MHz. When M66ENA is high, the SG2010 supports a functional CLK ranging from 33MHz to 66MHz.
RST_L	1	I	PCI platform reset input. When asserted low, resets the chip and, if the SG2010 is a root device, sends a maskable reset comma into the fabric. The SG2010 tristates all PCI outputs. If SG2010's central functions are enabled, the SG2010 drives AD[31:0], CBE_L[3:0] and PAR low during RSTO_L assertion.
RSTO_L	1	O	Reset output. Asserted by the SG2010 when it is reset by any reset mechanism. Asserted for the minimum PCI duration of 100 µsec. When the SG2010 is the central function, REQ64_L and AD[31:0], CBE_L[3:0] and PAR are driven by the SG2010 during the assertion of RSTO_L.
M66EN	1	I	PCI 66MHz enable. When sampled high, the SG2010 assumes that its PCI interface is operating between 33MHz and 66MHz and performs an additional clock divide to all time-sensitive functions based on the PCI clock (e.g. ROM interface, reset timer). When sampled low, the SG2010 assumes that its PCI interface is operating at 33MHz or below.
VIO	1	I	PCI I/O voltage bias. Tied to 3.3V or 5V to control the input voltage tolerance of the PCI receivers.

5.2 CompactPCI Hot Swap Signal Pins

Signal Name	Width	Type	Description
LSTAT	1	I	CompactPCI Hot Swap ejector handle switch status. Sampled by the SG2010 to determine when the ejector handle switch is open or closed, controlling the hot swap state machine. When a 1, the ejector handle is open. When a 0, the ejector handle is closed.
HS_LED	1	O	CompactPCI Hot Swap LED Control. Driven by the SG2010 and controlled by the hot swap function.
ENUM_L	1	BOD	CompactPCI Hot Swap interrupt. The SG2010 can be enabled to sample this signal and forward it through the event dispatcher, and can be enabled to assert this signal when it receives an event directed to the ENUM_L EMU address.
L64EN_L	1	I	CompactPCI Hot Swap local 64-bit extension enable. When the SG2010 samples this signal low, it enables the PCI 64-bit extension signals. When the SG2010 samples this signal high, the PCI 64-bit extension signal are only enabled if REQ64_L is sampled low during RST_L or LRST_L.
LRST_L	1	I	CompactPCI Hot Swap local reset. When asserted, the SG2010 performs a chip reset and, if the root, propagates a maskable reset into the fabric. The SG2010 tristates all PCI outputs as long as LRST_L is asserted.
BDSEL_L	1	I	CompactPCI Hot Swap board seated. When sampled high, the SG2010 does not respond to or initiate any PCI transactions (after completing any ongoing transactions). When sampled low, the SG2010 responds to and initiated PCI transactions normally.

5.3 PCI Arbiter and GPIO Signal Pins

Signal Name	Width	Type	Description
REQ_L[0]/AGNT_L	1	I	PCI arbiter input. If SG2010’s PCI bus arbiter is used, this signal is bus master 0’s PCI request input pin. If an external arbiter is used, this signal is SG2010’s PCI bus grant input pin.
GNT_L[0]/AREQ_L	1	TS	PCI arbiter output. If SG2010’s PCI bus arbiter is used, this signal is bus master 0’s PCI grant output pin. If an external arbiter is used, this signal is SG2010’s PCI bus request output pin.
REQ_L[6:1]	6	I	PCI arbiter dedicated request inputs. If SG2010’s PCI bus arbiter is used, these pins are PCI request inputs for up to seven PCI bus masters. If an external arbiter is used, these pins are ignored by the SG2010 and should be pulled up using external resistors.
GNT_L[6:1]	6	TS	PCI arbiter dedicated grant outputs. If SG2010’s PCI bus arbiter is used, these pins are PCI grant outputs for up to seven PCI bus masters. If an external arbiter is used, these pins are driven high by the SG2010.

Signal Name	Width	Type	Description
REQ_L[8:7]/GPIO[6,4]	2	TS	PCI arbiter shared request inputs/GPIO pins. If SG2010's PCI bus arbiter is used and these pins are enabled as arbiter pins, these pins are PCI request inputs for up to two PCI bus masters. If an external arbiter is used and the pins are enabled as arbiter pins, these pins are ignored by the SG2010 and should be pulled up using external resistors. If these pins are enabled as GPIO pins, they are controlled by the GPIO register function.
GNT_L[8:7]/GPIO[7,5]	2	TS	PCI arbiter shared grant outputs/GPIO pins. If SG2010's PCI bus arbiter is used and these pins are enabled as arbiter pins, these pins are PCI grant outputs for up to two PCI bus masters. If an external arbiter is used and the pins are enabled as arbiter pins, these pins are ignored by the SG2010 and are driven high. If these pins are enabled as GPIO pins, they are controlled by the GPIO register function.
GPIO[3:0]	4	TS	General purpose I/O pins. These pins are controlled by SG2010's GPIO register function.

5.4 ROM Interface Signal Pins

Signal Name	Width	Type	Description
PR_AD[0]/SR_DO PR_AD[1] PR_AD[2]/SKIPINS PR_AD[3]/LEDHM PR_AD[4]/PFN[0] PR_AD[5]/LOCKOUT PR_AD[6]/ARBEN PR_AD[7]/CFEN	8	TS	These pins are shared between the parallel ROM multiplexed address/data bus, serial ROM data output signal, and strapping pins. The parallel ROM uses these pins to drive 3 cycles of address data (24 bits total), followed by driving write data or receiving read data. The serial ROM uses PR_AD[0] to receive read data from the serial ROM. The strapping pins are sampled during reset to initialize SG2010 functionality. SKIPINS causes the hot swap controller to skip the insertion state and go directly to normal operation on power-up. LEDHM sets the mode for the LEDx[3:0] signals. PFN0 sets the value of this bit when the SG2010 is a root during hardware fabric enumeration. LOCKOUT controls link access to SG2010's registers after serial ROM preload when the SG2010 is a leaf. ARBEN enables or disables SG2010's PCI arbiter. CFEN enables or disables SG2010's PCI central functions. External pull-up or pull-down resistors are required on all PR_AD signal pins.
PR_RD_L	1	O	Parallel ROM read strobe. The SG2010 asserts this signal to enable read data to be driven on PR_AD[7:0} by the parallel ROM.
PR_WR_L	1	O	Parallel ROM write strobe. The SG2010 asserts this signal to indicate that the SG2010 is driving valid write data on PR_AD[7:0].
PR_CS_L/PR_RDY	1	TS	Parallel ROM chip select output/device ready input. When the parallel ROM interface is not in multi-function mode, the SG2010 drives this signal as an active low chip select signal for parallel ROM reads and writes. When the parallel ROM interface is in multi-function mode, the SG2010 samples this signal as a device ready signal. When deasserted during a read or write operation, it extends the read or write cycle for as long as it remains deasserted.
PR_ALE_L/SR_DI	1	O	Parallel ROM address latch enable/Serial ROM data input. When a parallel ROM operation is performed, enables the pipelined address to be externally latched. When a serial ROM operation is performed, the SG2010 drives the command, address, and if a write, write data onto this signal.

Signal Name	Width	Type	Description
PR_CLK/SR_CK	1	O	Parallel ROM address latch clock/Serial ROM clock. When performing a parallel ROM operation, this signal is driven by the SG2010 and used to clock the external address latches. When performing a serial ROM operation, this signal is driven by the SG2010 and used to clock the serial ROM.
SR_CS_L	1	O	Serial ROM chip select. When driven low by the SG2010, indicates the beginning of a serial ROM operation. The SG2010 drives this signal high when the operation is complete.

5.5 Configuration Signal Pins

Signal Name	Width	Type	Description
ROOT	1	I	When ROOT is high, the SG2010 is configured as a root. When ROOT is low, the SG2010 is configured as a leaf. This signal should either be driven, or tied high or low through an external resistor.
BRIDGE_EN	1	I	When BRIDGE_EN is high, SG2010’s Bridge function is enabled and the SG2010 generates and receives address-routed frames. When BRIDGE_EN is low, SG2010’s Bridge function is disabled and the SG2010 functions in Gateway-only mode, and cannot generate or receive address-routed frames.

5.6 Global PLL Signal Pins

Signal Name	Width	Type	Description
VDDG	1	P	VDD for 78MHz PLL
VSSG	1	P	VSS for 78MHz PLL
TSTCLKG	1	I	Reference clock used to bypass the 78 MHz PLL supplying the global clock. The 78MHz PLL is driven off of REFCLKL. Bypass mode is selected through TESTMODE pins.
PLLCLKGO	1	O	78MHz PLL output
RESET_PLL	1	I	On the negative edge of RESET_PLL, the 78MHz PLL re-locks to REFCLKL. If the PLL has previously locked to a different REFCLKL frequency, it will not automatically re-lock to a significant change in REFCLKL without a pulse on this pin. This pin is pulled low internally.

5.7 Link Interface Signal Pins

Signal Name	Width	Type	Description
TX0P[3:0]	4	LO	Link 0 LVDS transmit positive
TX0N[3:0]	4	LO	Link 0 LVDS transmit negative
TX1P[3:0]	4	LO	Link 1 LVDS transmit positive
TX1N[3:0]	4	LO	Link 1 LVDS transmit negative
RX0P[3:0]	4	LI	Link 0 LVDS receive positive
RX0N[3:0]	4	LI	Link 0 LVDS receive negative
RX1P[3:0]	4	LI	Link 1 LVDS receive positive
RX1N[3:0]	4	LI	Link 1 LVDS receive negative
REFCLKL	1	I	Reference clock for clock and data recovery (CDR) PLL
CTAP0[3:0]	4	I	Link 0 LVDS center taps for external reference voltages
CTAP1[3:0]	4	I	Link 1 LVDS center taps for external reference voltages
RESLO	1	I	LVDS 100Ω reference low – connects to RESLO through 100Ω 1% resistor.
RESHI	1	I	LVDS 100Ω reference high– connects to RESHI through 100Ω 1% resistor.
REF14	1	I	LVDS 1.4V reference
REF10	1	I	LVDS 1.0V reference
VDDA	2		Analog VDD
VSSA	2		Analog VSS

5.8 Link Interface Test Pins

Signal Name	Width	Type	Description
TSTCLKL	1	I	Bypass clock for CDR PLL
BYPASSL	1	I	Active high bypass enable for CDR
RESETTX	1	I	Tx clock divide reset for CDR PLL bypass
TSTSHFTLD	1	I	CDR test mode shift enable
TESTRST	1	I	CDR Bist reset
ECSEL	1	I	Manual CDR phase shift
ETOGGLE	1	I	CDR clock phase change
EXDNUP	1	I	CDR clock phase direction (1=up, 0=down)
TSTPHASE	1	I	Bypass phase control
LOOPBKEN	1	I	Loop back enable
TESTMUX[9:8]/ LED1_L[3:0]/TESTMUX[7:4] LED0_L[3:0]/TESTMUX[3:0]	10	TS	Test mode output port and transmit state LEDs. When the SG2010 drives LED state, a low level means the transmitter for the corresponding differential pair is synchronized and the Traffic Enable bit is set, and a high level means it is not synchronized. An oscillating level means the differential pair is synchronized but the Traffic Enable bit is not set.

5.9 Test Signal Pins

Signal Name	Width	Type	Description
TESTMODE[3:0]	4	I	Defines the following test modes in the SG2010: 0h:Functional/JTAG 1h:Functional/PLL bypass 2h:SCAN 3h:IDDQ 4h:Tristate 5h:Global PLL test 6h:Reserved (GPIO mode) 7h:LED test 8h:Reserved 9h:Reserved Ah:CDR Test 1 Bh:CDR Test 2 Ch:Reserved Dh:Reserved Eh:Reserved Fh:Reserved
TCK	1	I	JTAG clock
TDI	1	I	JTAG data in
TDO	1	O	JTAG data out
TMS	1	I	JTAG mode select
TRST_L	1	I	JTAG reset
SCAN_ENA	1	I	Scan enable input

5.10 Other Pins

Signal Name	Width	Type	Description
RESERVED[2:0] RESERVED[4]	5	TS	Reserved for future use.

Signal Pin List

The SG2010 is packaged in a 272-pin Plastic Ball Grid Array (PBGA) package. The pinout of the SG2010 is shown in Table 6–1.

Table 6–1 Pin List By Location

Pin	Signal Name	Type
B1	tstshftld	I
C2	ecsel	I
D2	etoggle	I
D3	exdnup	I
E4	tstphase	I
C1	testmux[9]	IO
D1	testmux[8]	IO
E3	led1_l[3]/testmux[7]	IO
E2	led1_l[2]/testmux[6]	IO
E1	led1_l[1]/testmux[5]	IO
F3	led1_l[0]/testmux[4]	IO
G4	led0_l[3]/testmux[3]	IO
F2	led0_l[2]/testmux[2]	IO
F1	led0_l[1]/testmux[1]	IO
G3	led0_l[0]/testmux[0]	IO
G2	reserved[4]	IO
G1	PLL_Reset	IO
H3	reserved[2]	IO
H2	reserved[1]	IO
H1	reserved[0]	IO
J4	enum_l	O
J3	hs_led	O
J2	lstat	I
J1	lrst_l	I

Table 6–1 Pin List By Location

K2	l64en_l	I
K3	bdsel_l	I
K1	root	I
L1	bridge_en	I
L2	rsto_l	O
L3	scan_ena	I
M1	gnt_l[0]/areq_l	IO
M2	req_l[0]/agnt_l	IO
M3	gnt_l[1]	IO
M4	req_l[1]	IO
N1	gnt_l[2]	IO
N2	req_l[2]	IO
N3	gnt_l[3]	IO
P1	req_l[3]	IO
P2	gnt_l[4]	IO
R1	req_l[4]	IO
P3	gnt_l[5]	IO
R2	req_l[5]	IO
T1	gnt_l[6]	IO
P4	req_l[6]	IO
R3	gnt_l[7]/gpio[5]	IO
T2	req_l[7]/gpio[4]	IO
U1	gnt_l[8]/gpio[7]	IO
T3	req_l[8]/gpio[6]	IO
U2	gpio[3]	IO
V1	gpio[2]	IO
T4	gpio[1]	IO
U3	gpio[0]	IO
V2	trst_l	I
W1	tck	I
V3	tms	I
W2	tdo	O
Y1	tdi	I
W3	inta_l	O
Y2	intb_l	O
W4	intc_l	O
V4	intd_l	O
U5	rst_l	I
Y3	clk	I
Y4	pme_l	O
V5	ad[31]	IO

Table 6–1 Pin List By Location

W5	ad[30]	IO
Y5	ad[29]	IO
V6	ad[28]	IO
U7	ad[27]	IO
W6	ad[26]	IO
Y6	ad[25]	IO
V7	ad[24]	IO
W7	cbe_l[3]	IO
Y7	idsel	I
V8	ad[23]	IO
W8	ad[22]	IO
Y8	ad[21]	IO
U9	ad[20]	IO
V9	ad[19]	IO
W9	ad[18]	IO
Y9	ad[17]	IO
W10	ad[16]	IO
V10	cbe_l[2]	IO
Y10	frame_l	IO
Y11	irdy_l	IO
W11	trdy_l	IO
V11	devsel_l	IO
Y12	stop_l	IO
W12	perr_l	IO
V12	serr_l	O
U12	par	IO
Y13	cbe_l[1]	IO
W13	ad[15]	IO
V13	ad[14]	IO
Y14	ad[13]	IO
W14	ad[12]	IO
Y15	ad[11]	IO
V14	ad[10]	IO
W15	m66en	I
Y16	ad[9]	IO
U14	ad[8]	IO
V15	cbe_l[0]	IO
W16	ad[7]	IO
Y17	ad[6]	IO

Table 6–1 Pin List By Location

V16	ad[5]	IO
W17	ad[4]	IO
Y18	ad[3]	IO
U16	ad[2]	IO
V17	ad[1]	IO
W18	ad[0]	IO
Y19	ack64_l	IO
V18	req64_l	IO
W19	cbe_l[7]	IO
Y20	cbe_l[6]	IO
W20	cbe_l[5]	IO
V19	cbe_l[4]	IO
U19	par64	IO
U18	ad[63]	IO
T17	ad[62]	IO
V20	ad[61]	IO
U20	ad[60]	IO
T18	ad[59]	IO
T19	ad[58]	IO
T20	ad[57]	IO
R18	ad[56]	IO
P17	ad[55]	IO
R19	ad[54]	IO
R20	ad[53]	IO
P18	ad[52]	IO
P19	ad[51]	IO
P20	ad[50]	IO
N18	ad[49]	IO
N19	ad[48]	IO
N20	ad[47]	IO
M17	ad[46]	IO
M18	ad[45]	IO
M19	ad[44]	IO
M20	ad[43]	IO
L19	ad[42]	IO
L18	ad[41]	IO
L20	ad[40]	IO
K20	ad[39]	IO
K19	ad[38]	IO

Table 6–1 Pin List By Location

K18	ad[37]	IO
J20	vio	I
J19	ad[36]	IO
J18	ad[35]	IO
J17	ad[34]	IO
H20	ad[33]	IO
H19	ad[32]	IO
H18	testmode[3]	I
G20	testmode[2]	I
G19	testmode[1]	I
F20	testmode[0]	I
G18	pr_ad[7]/CFEN	IO
F19	pr_ad[6]/ARBEN	IO
E20	pr_ad[5]/LOCKOUT	IO
G17	pr_ad[4]/PFN[0]	IO
F18	pr_ad[3]/LEDHM	IO
E19	pr_ad[2]/SKIPINS	IO
D20	pr_ad[1]	IO
E18	pr_ad[0]/SR_DO	IO
D19	pr_rd_l	O
C20	pr_wr_l	O
E17	pr_cs_l	IO
D18	pr_ale_l	O
C19	pr_clk	O
B20	sr_cs_l	O
C18	tstclkg	I
B19	pllclkgo	O
A20	testrst	I
A19	rx0p[0]	I
B18	rx0n[0]	I
B17	ctap0[0]	I
C17	rx0p[1]	I
D16	rx0n[1]	I
A18	ctap0[1]	I
A17	ctap0[2]	I
C16	rx0p[2]	I
B16	rx0n[2]	I
A16	ctap0[3]	I
C15	rx0p[3]	I

Table 6–1 **Pin List By Location**

D14	rx0n[3]	I
B15	rx1p[0]	I
A15	rx1n[0]	I
C14	ctap1[0]	I
B14	rx1p[1]	I
A14	rx1n[1]	I
C13	ctap1[1]	I
B13	rx1p[2]	I
A13	rx1n[2]	I
D12	ctap1[2]	I
C12	rx1p[3]	I
B12	rx1n[3]	I
A12	ctap1[3]	I
B11	tx0p[0]	O
C11	tx0n[0]	O
A11	tx0p[1]	O
A10	tx0n[1]	O
B10	tx0p[2]	O
C10	tx0n[2]	O
A9	tx0p[3]	O
B9	tx0n[3]	O
C9	reslo	I
D9	reshi	I
A8	ref14	I
B8	ref10	I
B7	tx1p[0]	O
A6	tx1n[0]	O
C7	tx1p[1]	O
B6	tx1n[1]	O
D7	tx1p[2]	O
C6	tx1n[2]	O
B5	tx1p[3]	O
A4	tx1n[3]	O
C4	tstclkl	I
B3	refclkl	I
B2	bypassl	I
A2	resettx	I
C3	loopbken	I

Index

Acronyms

BAR	Base address register
Cfg	Configuration
CoS	Class of service
CSR	Control and status register
DAC	Dual address cycle
ECP	Extended capabilities port
ELP	Extended function list pointer
EMU	Event Message Unit
FID	Fabric ID
GPIO	General-purpose I/O
H/W	Determined by hardware
HP	High priority
MWI	Memory write and invalidate
P2P	PCI–to–PCI
PCI	Peripheral component interconnect
PFN	Parallel fabric number
SAC	Single address cycle
SFC	StarFabric component
SGF	Software generated frame
SGT	Software generated transaction
VPD	Vital product data
W1TS	Write 1 to set

Glossary

address routing	A mechanism used for routing frames through a fabric based on address decoding of the frame's address field at each node. PCI frame routing through a PCI hierarchy uses address routing.
bridge	An edge node that provides protocol translation; for example, a bridge between the fabric and a PCI bus.
bundled link	A bundled link has multiple link instances.
bundled port	A port that aggregates more than one link.
channel	A partitioned address range, used for address and path protection and address translation at edge nodes, and for StarFabric register space at all nodes.
Channel 255	A dedicated channel used for register accesses for all StarFabric components.
chunk	A potentially non-contiguous data stream of indeterminate size whose transmission integrity is assured by sequence numbers and CRC.
DAC	Dual address cycle. A 64-bit memory region. *See also* **SAC**.
differential pair **differential signal pair**	A pair of signal wires that connect a differential output buffer (Tx+ and Tx–) to a differential input buffer (Rx+ and Rx–).
disjoint route	A secondary route that shares no common switches with the primary route is said to be a disjoint route.
downstream	Used with transaction flow, frame direction, port, PCI bus, or PCI device to indicate flow away from the root.
edge node	Any node that is not a switch – either a bridge or a StarFabric native device.
fabric	The switched–serial interconnect using StarFabric protocol, which includes switches and edge nodes, and the links that connect them.
frame header	The first 12 bytes internal to a frame. The header bytes provide frame management and are not part of the data payload.

header	*See* **frame header** *and* **StarFabric component header**.
hybrid node	An edge node that is capable of receiving frames from one node and forwarding them to another.
initiator	A device that initiates a PCI transaction on a PCI bus.
input port	The port used by a node for frame reception. Used in the context of a single path or frame.
leaf	Any edge node that is not a root node. Typically used in reference to edge nodes in a PCI hierarchy.
line	A unit of frame size. The smallest frame is one line, and all frames sizes are on single line granularities. A line is 16 bytes, or four Dwords.
line buffer	Storage in a node used to hold a received frame line before it is transmitted.
line credit	Allocates or frees buffer space, in line granularity, in a node.
link	A physical connection between nodes consisting of all the elements necessary for two nodes to communicate. A link includes the link interface in each node and the differential signal pairs that connect them. A link reference is used for operations involving line credits, fabric enumeration, CRC calculation, frame transmission, and frame reception.
link interface	The functional block within a node that is composed of a link transmitter and a link receiver.
link number	Refers to the absolute link numbers of a node’s links.
link partner	The node that is connected to the StarFabric component in question, through a particular link, port, or differential pair.
link receiver	The subsection of a link interface that receives a frame. The link receiver includes the differential input buffers, clock recovery, serial-to-parallel converters, 8b/10b decoder, CRC logic, framing logic, and synchronization logic.
link transmitter	The subsection of a link interface that transmits a frame. The link transmitter includes the synchronization logic, framing logic, CRC logic, 8b/10b encoder, parallel-to-serial converters, and differential output buffers.
minimum turnaround latency	The minimum amount of time between the sending of a frame and the return of the line credits from the receiver of that frame.
multicast routing	A mechanism used for routing frames from a single origin to multiple edge nodes.
next turn	The turn specification for the next node in a path. Used for line credit operations and path routing.

node	Generic name for any edge node or switch that supports the StarFabric. Also referred to as a StarFabric component.
NPND	Nonprotocol native device.
origin	A node that is the generator of a frame. In the general case, the origin is an edge node, although there are specific cases where a switch can be an origin.
output port	The port on a node by which a frame is sent. Used in the context of a single path or frame.
path	The StarFabric specification for the position of a terminus relative to the origin in a fabric. The path is a field in the frame header and is used by nodes to direct a path-routed frame through the fabric.
path transform	The invert–and–reverse operation performed on a path specification to obtain the path to the origin of the frame.
path routing	A mechanism used for routing frames through a fabric based on a relative path from the origin to the terminus.
peer–to–peer	Refers to the flow of frames from one leaf node and terminating on another leaf node.
PND	Protocol native device.
port	A logical connection representing the set of links that connects one node to another node. This connection representation is used for operations involving frame routing (address decoding, turn counts, multicast groups).
receiver	A node that is receiving a frame from another StarFabric component. Used in the context of a single frame transmission or a specific path.
root	The node that initiates mesh/fabric enumeration, and, for PCI-compatible fabrics, the node closest to the host processor in the PCI hierarchy.
route	The trail from an origin to a terminus. A particular route is specified by a path in StarFabric protocol.
SAC	Single address cycle. A 32-bit memory region. *See also* **DAC**.
SPNC	*See* **StarFabric native component**.
StarFabric component header	The set of standard registers implemented by all StarFabric components. The StarFabric component header is mapped at offset 0 of Channel 255 address space.
StarFabric native component	An edge node that interfaces the StarFabric directly to a native function.

StarFabric component	Any edge node or switch that supports the StarFabric. Also referred to as a node.
StarFabric or SF	The switched–serial protocol described in this document.
striping	The technique for increasing bandwidth by using multiple parallel ports to transfer a unit of data. For example, initial StarFabric devices can bundle up to four 622Mbps differential pairs into a 2.5Gbps full duplex link, for 5Gbps of total bandwidth.
switch	A node that receives a frame from one node and forwards the frame to another node.
target	A device that is the target of a PCI transaction on a PCI bus. Also referred to as a PCI target.
terminus	A node that is the intended end receiver of a frame in the fabric. In the general case, the terminus is an edge node, although there are specific cases where a switch can be a terminus.
thread	A physical link within a bundle.
thread number	Threads within a bundle are numbered for dependency and error reporting purposes. Thread numbers are bundle-specific.
transmission medium	The transmission medium is the physical material used to transport signals between two nodes. Typical transmission mediums include copper PCB etch, and copper conductors in a cable.
transmitter	A node that is sending a frame to another StarFabric component. Used in the context of a single frame transmission or a specific path.
turn	Refers to the relative direction a frame takes when traveling through a switch; that is, the position of the output port relative to the input port of a switch.
turn count	Indicates the number of valid turns in a path specification. When the path specification is in a frame header, indicates the number of valid turns that have been taken at that point in the route.
upstream	Used with transaction flow, frame direction, port, PCI bus, or PCI device to indicate flow towards the root.