Advanced SerialRAID Adapters

Technical Reference

SA33-3286-02

Advanced SerialRAID Adapters

Technical Reference

SA33-3286-02

Note!

Before using this information and the product it supports, be sure to read the general information under "Appendix B. Notices" on page 333.

Third Edition (September 2000)

This major revision supersedes SA33-3286-01. Technical changes are shown by a vertical line to the left of each change.

Contents

Figures

Conventions

Bits

The Advanced SerialRAID Adapter use the standard convention for numbering the bits within bytes and words. Bit 0 is the least-significant bit; the number of the most-significant bit is 1 less than the width of the data.

Bit values are represented like this: 010b

Hexadecimal values are represented like this: 7Ah

Bytes

Except as noted below, the adapters’ host interface uses the Little-endian convention, that is, it assumes that the least-significant byte of a number or an address is stored at the lowest byte address.

The Power processors in pSeries and RS/6000 servers use the Big-endian convention. Therefore, AIX device drivers must take specific action to reverse the byte order that is naturally generated by the processor. PowerPC processors can operate in either Big-endian or Little-endian mode.

When using an SSA adapter in SCSI pass-through mode, it is important to note that parallel SCSI, and hence SSA-SCSI, sends the most-significant byte of a number first. This means that numbers in command-descriptor blocks, sense data and mode parameters appear in Big-endian format in memory.

Words

In an Advanced SerialRAID Adapter, a **word** is 4 bytes.

Registers

All register bits are read/write unless explicitly noted in the description of a bit.

Serial links

When an SSA adapter transfers information over a serial link, the bytes are normally sent and received in strict order of ascending storage addresses. This guarantees that customer data can be retrieved correctly when an SSA disk drive is interchanged between different host systems.

Register bit ranges

In this book we use a dash (–) to indicate a range of bits in a register. For instance, 15–10 indicates bits 15, 14, 13, 12, 11, and 10 of a register; 2–0 indicates bits 2, 1 and 0.

Chapter 1. Description

IBM Servers

The Advanced SerialRAID Adapter can be used on IBM @server pSeries systems, SP/2, and PC-based systems. In this book we indicate where differences occur. For the most part the functioning of the adapter is consistent across all using systems. Attention is drawn to differences where they exist.

Introduction to the Adapter

The Advanced SerialRAID Adapter is a PCI bus-master adapter that serves as the interface between systems using the Peripheral Component Interconnect (PCI) architecture and disk drives using the Serial Storage Architecture (SSA). SSA is an open standard that defines a high-performance serial interface for storage devices. It retains the SCSI-2 commands, queuing model and sense bytes.

| If the code in the Advanced SerialRAID Adapter is at, or higher than, level 5000, the
| adapter is known as an Advanced SerialRAID Adapter Plus and provides addition
| functions. The additional functions supported by the Advanced SerialRAID Adapter Plus
| are shown in this manual as being available only at code level 5000 and higher.

| The adapter provides high-performance implementation of non-RAID disks, RAID-0, and
| RAID-5 arrays, all with an optional non-volatile fast write cache (FWC). RAID-1 and
| RAID-10 arrays are supported if the code level is at, or higher than, level 5000.
| High-availability clusters with no single point of failure can be configured with up to 8
| adapters on an @server system, or up to 2 adapters on PC Server systems, if all the
| disks are non-RAID and not configured with FWC. If the code level is lower than level
| 5000, the cluster can be up to 2 adapters if any disks are configured as members of a

RAID array that is not configured with FWC. If the code level is at, or higher than, level 5000, the cluster can be up to 2 adapters if any disks are configured as members of a RAID-1, RAID-5, or RAID-10 array that can be configured with FWC.

For pSeries, @server, and SP/2 systems, code at or above level A000 is available. Code level A000 and above provides the following additional functions:

1. 3-Way Copy. This is only possible with AIX Version 4.3.3 or later.
2. Improve performance of sequential Reads on RAID-1 and RAID-10.
3. Optimise partial stripe Writes by reading strips not written and then issuing a Write to the full stripe.
4. Improve error handling to reduce adapter resets due to timeouts.

The adapters each provide 4 SSA ports that operate as 2 dual-port nodes for the attachment of storage devices such as hard disk drives. Each port operates at 20 or 40 MB/s full-duplex communication, automatically negotiating the rate with the remote node, using point-to-point copper cables up to 25 meters long. As an alternative to copper cables, fiber optic cables can be used to link SSA nodes. Nodes linked by fiber optic cables can be up to 2.4km (7874ft) apart with multi-mode fibers or 10km (32800 ft) apart with single-mode fibers. Fibre-Optic Extenders, which are features of SSA units, connect the fiber optic cables to the SSA nodes.

Each of the 2 pairs of SSA ports can attach up to 48 dual-port devices in a closed loop. If the loop is broken by a fault, the two ports continue to access the devices using the remaining connections as a string; however, it is not intended that the devices should be configured as a string initially. These SSA features support fault-tolerant applications.

SSA Ports

- 20 or 40 MB/s for each port, automatically negotiated with the remote node:
  - Mixed speeds in the same loop. (This is a migration aid.)
  - Full-duplex communication with frame multiplexing and spatial reuse.
  - Up to 25 meters for each link using copper cable.
  - Up to 10 kM at reduced bandwidth with an optical extender.
  - Hot-plugging. (Adapters and devices.)
- The Advanced SerialRAID Adapters comply with the extended SSA-IA/95PH and SSA-IA/95SP to support 40 MB/s links:
  - 40 MB/s electrical specifications according to SSA-PH2.
  - Speed negotiation process for 20 or 40 MB/s according to SSA-TL2.
  - Speed renegotiation when the link error rate lies outside certain thresholds.
  - Extensions to certain transport-layer messages.
- 4 ports that operate as 2 dual-port nodes:
  - Each dual-port is an independent initiator with its own Unique_Id.
- 2 SSA loops:
  - No single point of failure. (Each loop can fall-back to a string.)

- Automatic network configuration.
- Automatic selection of the shortest available path to each node.

Control and Data Store

- 64 MB Synchronous DRAM, with ECC protection. (Field replaceable SDRAM.) On pSeries, RS/6000, and SP/2 systems this can be increased to 128 MB to support full 32 MB write cache when in a 2-way cluster.

Non-volatile Write Cache

- PCI daughter card. A customer installable option. The daughter card is removable without data loss in the event of the base adapter card failing.
- 32 MB low power EDO DRAM.
- 7 days minimum data retention using a field-replaceable, internal, nickel-cadmium secondary battery.
- Error detection and recovery. A horizontal parity check on each word using a redundant memory module plus vertical CRC and a time-stamp for each block, is provided.

Firmware

The Advanced SerialRAID Adapter uses the Independent Packet Network (IPN) firmware base. In addition to non-RAID disks, it can support advanced functions such as 2-way RAID and a write cache. Each of these advanced functions is implemented as an independent filter. The output of a filter is a logical disk; the input may be another logical or physical disk or disks.

Under control of the adapter configuration utility, filters may be cascaded as shown in Figure 1

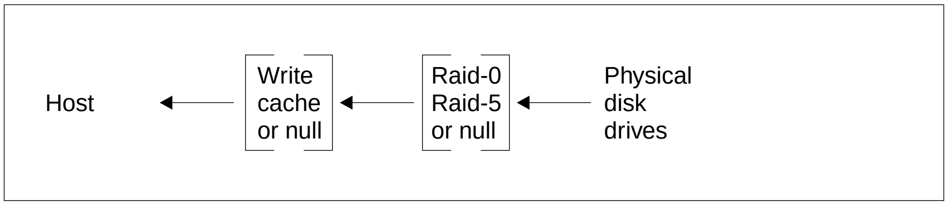


Figure 1. Firmware Filters

The functions of the individual filters are described below.

The performance highlights in this section are with the adapter operating close to saturation. Maximum data rates are subject to the particular implementation of the host PCI bus. When configuring a complete storage sub-system a margin should be allowed for transient peaks in I/O activity.

Non-RAID Disks

- 8500 operations per second. (4 KB transfers, 30% writes, no write cache.)
- 85 MB/s sustained read or write bandwidth. (64 KB transfers, no write cache.)

RAID-0 Filter

- 2 to 16 members per array.
- 4 KB to 256 KB strips in 4 KB increments.
- 6,500 operations per second. (4 KB transfers, 30% writes, no write cache.)
- 80 MB/s sustained read or write bandwidth. (64 KB transfers, 64 KB strips, no write cache.)

RAID-1 Filter

RAID-1 is supported if the code level is at, or higher than, level 50.

- 2 member disks per array.
- Mirrored copies of data held on each member.
- Tolerant to a single medium error or drive failure.
- Data scrubbing. A background process in the adapter periodically verifies that all data can be successfully read from the disk drives.
- Optional global hot spare or spares. (A spare drive may be available to more than one array.)
- Concurrent rebuild with programmable priority.
- Option for two adapters per array with duplex copies of non-volatile memory. (Dual-active with shared disks.)
- Members can be configured to allow operation to continue when an entire site fails or loses power.
- 4,700 operations per second. (4 KB transfers, 30% writes, no write cache.)
- 60 MB/s sustained read or write bandwidth. (64 KB transfers, no write cache.)
- Read performance is enhanced because there is a choice of members that can be accessed for the data. Write performance is reduced because 2 copies of the data have to be written.

RAID-5 Filter

- 3 to 16 member disks per array containing data and distributed parity.
- 32 or 64 KB strips (plus 16 KB strips on PC Servers).
- 4 or 5 stripes per stretch.
- Tolerant to a single media error or drive failure.
- Non-volatile memory to log parity updates in case of power failure during writes. Automatic parity synchronization when power is restored.
- Data scrubbing. A background process in the adapter periodically verifies that all data and parity can be successfully read from the disk drives.
- Optional global hot spares. A spare drive may cover more than one array.
- Concurrent rebuild with programmable priority.

- Option for 2 (dual-active) adapters per array with duplex copies of non-volatile memory.
- Lazy parity. Parity is updated after completing the host transaction.
- Integrated read cache for data and parity. This is distinct from the write cache filter described below.
- 9,000 array operations per second. (4 KB transfers, 100% read hit, no write cache, single adapter.)
- 2,700 array operations per second. (4 KB transfers, 30% writes, no write cache, single adapter.)
- 80 MB/s sustained bandwidth. (6+p arrays, full-stripe transfers, 30% writes, no write cache, single adapter.)

RAID-10 Filter

RAID-10 is supported if the code is at, or higher than, level 5000. Some documentation refers to this type of RAID as RAID 0+1; the term RAID-10 is used in this document.

- 4 to 16 member disks per array.
- Mirrored copies of data held on each pair of member disks.
- 16, 32, or 64 KB strips.
- Tolerant to a single medium error or drive failure.
- Member disks can be configured to allow operation to continue when an entire site fails or loses power.
- Data scrubbing. A background process in the adapter periodically verifies that all data can be successfully read from the disk drives.
- Optional global hot spare or spares. (A spare drive may be available to more than one array.)
- Concurrent rebuild with programmable priority.
- Option for two adapters per array with duplex copies of non-volatile memory. (Dual-active with shared disks.)
- 5,000 operations per second. (4 KB transfers, 30% writes, no write cache.)
- 60 MB/s sustained read or write bandwidth. (64 KB transfers, no write cache.)
- Read performance is enhanced because there is a choice of members that can be accessed for the data. Write performance is reduced because 2 copies of the data have to be written.

Write Cache Filter

The non-volatile write cache is implemented using a daughter card:

- Redundant copies of write data to ensure no single point of data loss. The Advanced SerialRAID Adapter makes 1 non-volatile copy in the fast write cache plus 1 volatile copy in SDRAM. Each copy is fully self-describing. In an advanced function cluster (supported by code level 50 and above), the other adapter makes a further copy in its SDRAM.
- Fast write. The Advanced SerialRAID Adapter completes the host write transaction immediately after all copies of the write data have been transferred into the cache.

- Write pre-empt. When a block that is already in the write cache is written again, only a single destage is performed.
- Write blocking. When a block is written whose LBA immediately follows a block that is already in the write cache, the writes are merged into a single destage.
- Least-Recently-Used (LRU) data is destaged when the cache is more than 70% full. Data is also destaged when it has been in the cache for more than 2 minutes. For RAID-5, the Advanced SerialRAID Adapter will destage full stripes when possible. Depending on the host operating system, outstanding data is also destaged when the system is shut down normally.
- Supports Non-RAID, RAID-0, and RAID-5 with a single adapter if the code is below level 50. Supports Non-RAID, RAID-1, RAID-5, and RAID-10 in advanced function clusters if the code is at level 50 or higher.
- Selective caching by logical disk, LBA range, and transfer length.
- 4,500 operations per second. (Non-RAID, 4 KB random transfers, 30% writes.)
- 70 MB/s sustained bandwidth. (Non-RAID, 64 KB sequential transfers, 30% writes.)

3–Way Copy

3-Way Copy allows a user to add a third copy to an existing RAID-1 or RAID-10 array. A background process then copies data from the original copies to the new copy in a similar manner to an array rebuild. During the copy process, read and write operations may still continue to the original copies. Write operations that are submitted to the array during the process of building the third copy will update all three copies of the data.

Once the background copy process has completed, all write operations continue to update all three copies of the data. At any time after the copy process has completed, the third copy may be uncoupled from the original array to form an independent resource. To perform the uncouple of the 3rd copy, scripts are provided to synchronize and stop I/O operations and flush any fast write data to disk. This ensures that any data cached is flushed to disk. As soon as the uncouple has occurred, operations may be resumed to the arrays.

Before this RAID copy resource can be used, certain operating system metadata needs to be modified to make this second copy non-identical, otherwise the operating system would see two identical resources with identical names.

At the time the third copy is uncoupled from the RAID-1 or RAID-10 array, the user is effectively taking a snapshot of the resource. This snapshot copy would then typically be used to perform a backup or to test some new application.

Once the user has finished with the copy, those disks may be re-attached to another RAID-1 or RAID-10 array and the process of copying the array data onto these disks repeated.

Clusters

A cluster is a configuration with multiple adapters. This permits high-availability systems with no single point of failure. It also allows higher overall through-put by sharing the workload between several hosts and adapters.

The Advanced SerialRAID Adapter supports two types of cluster:

- Non-RAID only. A non-RAID cluster may have up to 8 adapters in each loop on a pSeries, RS/6000, or SP/2 based system or 2 adapters on a PC Server system.
- Advanced-function. An advanced-function cluster supports non-RAID, RAID-5, or both, without write cache if the code is below level 50. If the code level is at, or higher than, level 50, an advanced function cluster supports non-RAID, RAID-1, RAID-5, or RAID-10 and any of these can have a write cache. RAID-0 is not supported. An advanced-function cluster is limited to 2 adapters in each loop.

Table 1. Maximum numbers of Adapters in a Cluster

Type of Resource	Maximum Number of Adapters in Cluster	
	Code level < 50	Code ≥ 50
Non-RAID	8 (non-PC Systems) 2 (PC Systems)	8 (non-PC Systems) 2 (PC Systems)
RAID-0	1	1
RAID-1	Not supported	2
RAID-5	2	2
RAID-10	Not supported	2
Fast Write Cache	1	2

An example of an advanced-function cluster with non-RAID drives and a RAID-5 array is shown in Figure 2.

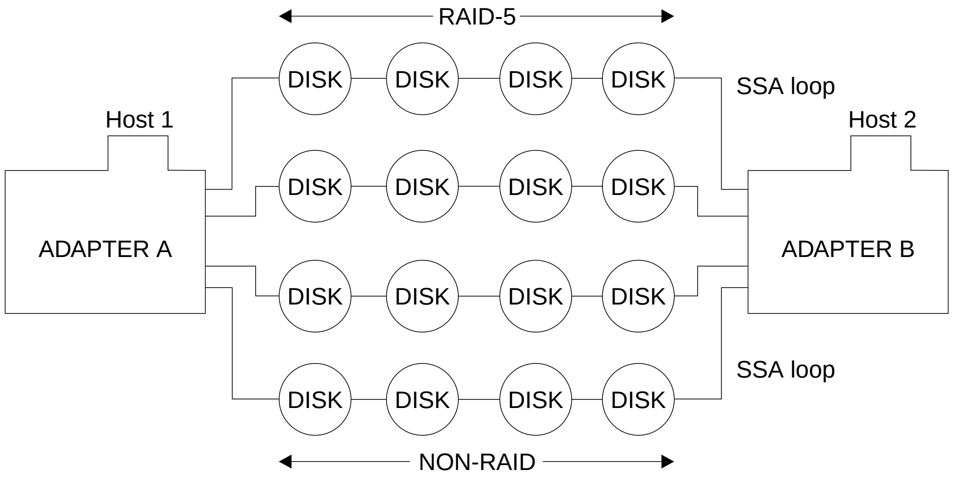


Figure 2. 2–way Advanced Function Cluster

- There are two loops. Each loop contains both adapters and it also contains non-RAID drives, RAID-5 arrays, or both. All of the members of a RAID-5 array (including hot spares) must be in the same loop.

- Each physical disk drive is managed by both adapters.
- The two adapters are dual-active. They share any RAID-5 arrays, that is both adapters can simultaneously access the same array.
- The RAID-5 filters in both adapters exchange locks through the SSA loops to ensure logical consistency of the arrays. They also maintain duplex copies of non-volatile memory to protect against an adapter failure.

The Advanced SerialRAID Adapter provide the following additional functions for clusters:

- Adapter fail-over. The device driver supports automatic fail-over between two redundant adapters in the same pSeries, RS/6000, or SP/2 host system.
- Fencing. Access to each logical disk can be allowed or disallowed for any subset of the adapters.
- SSA Target mode. This provides host-to-host communication, for example to support Volume Status Change messages for LVM synchronization and host heartbeat messages. The device driver makes this appear like SCSI target mode but it is actually implemented with an IPN protocol.

  Note: SSA Target mode is not intended for performance-intensive applications.

Some performance highlights for clusters

Table 2. Cluster performance. (Total for all adapters with 4 KB transfers, 30% writes and no write cache.)

Configuration	Non-RAID	RAID-1	RAID-5	RAID-10
2-way advanced-function cluster	16,000 ops/s	7,000 ops/s	4,000 ops/s	7,500 ops/s
4-way cluster for non-RAID only	30,000 ops/s	n/a	n/a	n/a
8-way cluster for non-RAID only	60,000 ops/s	n/a	n/a	n/a

Configurations

An example of an SSA subsystem that can be attached to a Advanced SerialRAID Adapter is the **7133 Serial Disk System**. Each 7133 unit contains up to 16 SSA disk drives with fault-tolerant power and cooling. It is available either as a 19-inch rack-mounted unit or as a deskside unit.

Adapter Functions

The principal functions of the Advanced SerialRAID Adapter are:

- The adapter performs a power-on self-test (POST) to verify correct operation of the hardware.
- The adapter configures the SSA network. It can act as the master node if required.

- When interrupted by the host processor, the adapter fetches IPN transactions by Direct Memory Access (DMA) from host memory.
- The adapter translates each transaction into SCSI commands and issues them to the addressed device over a serial link. A pass-through mode is also provided to allow any SCSI command to be issued.
- When requested by a device, the adapter fetches write data from host memory by DMA and transmits it to the device. Similarly the adapter receives read data from the device and stores it in host memory by DMA.
- For disk drives that are not in an array, data is transferred between the host memory and the devices through SDRAM. For RAID-5, data is transferred via the data buffer; subsequent reads might be satisfied from the data buffer without a further operation to the device. The adapter can scatter or gather the data to or from noncontiguous regions of host memory.
- The adapter receives SCSI status from the device. If there is an error the adapter issues a SCSI Request Sense command to the device and may then attempt to recover the error. In all cases the adapter interrupts the host processor to present the result of the transaction and to log errors if appropriate.

Supported Standards

The Advanced SerialRAID Adapter implements the standards described in the following documents:

- *PCI Local Bus Specification*, production version, revision 2.1.
- *Serial Storage Architecture, 1995 Physical (SSA-IA/95PH)*, October 1995.
- *Serial Storage Architecture, 1995 SCSI-2 Protocol 9SSA-IA/95SP)*, October 1995.
- *Small Computer System Interface - 2 (SCSI-2)*, X3.131.199X, Revision 10m.

Introduction to the Independent Packet Network (IPN)

The device drivers and adapter communicate with each other by means of a logical client-server network called an Independent Packet Network (IPN).

IPN is a logical network of **services**. A client can access a service by specifying its address in the IPN network, without being concerned where the service is physically located. In IPN terminology, the client is a **master** and the service is a **slave**.

The unit of work in IPN is a **transaction**. The routing layer of IPN establishes a connection between the master and slave for the duration of each transaction. A master may queue multiple transactions in the same slave. However, the slave can execute the transactions in any order it chooses and can even execute several transactions concurrently.

An IPN **node** is a hardware unit that runs the IPN kernel; a host system or the adapter are examples of nodes. In addition to network routing, the IPN kernel also performs such tasks as scheduling, memory management, and timer functions.

The adapter provides a **disk service** to give basic read/write access to each attached disk drive. Additional services can be added, such as a RAID service.

The host device driver is an IPN master and also provides an **error logger**, which is a service for logging subsystem errors.

Every IPN node also contains a **registry** service. The registry keeps a list of all services running on its node and all other nodes that are directly accessible through a gateway on that node. The registry also forwards errors detected by the services running on its node to the error logger.

IPN spans the device driver and the adapter. IPN uses a **gateway** to cross a physical interface such as the PCI interface. The gateway is transparent to the master and slave and it incorporates the specific features of the physical interface.

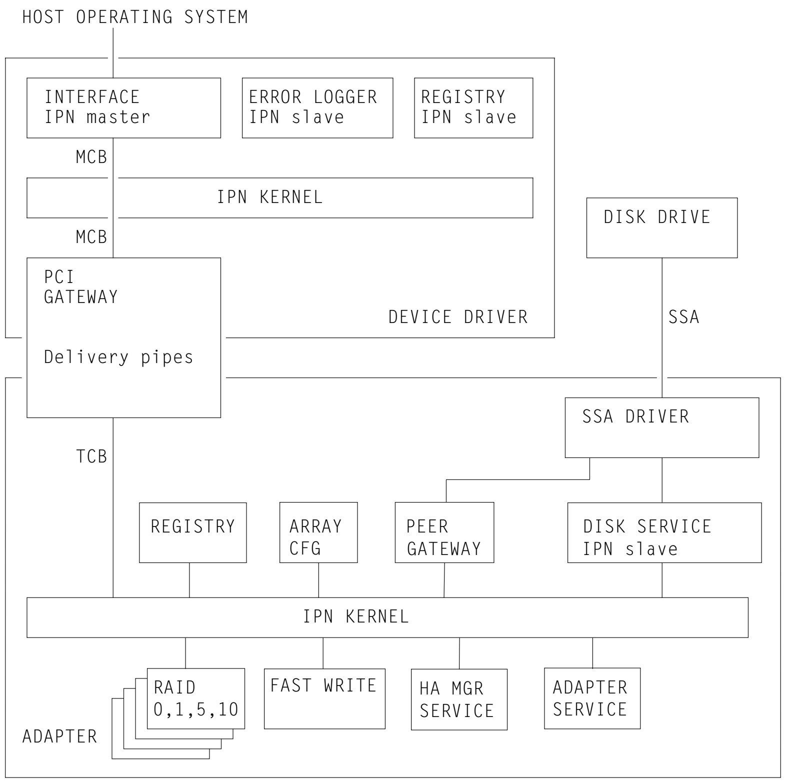


Figure 3. IPN Components

The Advanced SerialRAID Adapter contains a PCI gateway, a disk service, a registry service, an SSA driver, the IPN kernel, and a service for each RAID function, as shown in Figure 3. A typical transaction to read data from a RAID-5 array would be processed as follows:

1. The device driver contains a master process that generates IPN transactions. The master calls the host IPN kernel with a pointer to a **master control block (MCB)** for the transaction. The MCB is addressed to the RAID-5 service.
2. The host IPN kernel calls the PCI gateway with a pointer to the MCB.
3. The host side of the PCI gateway creates a **gateway transaction control block (GTCB)** in host memory. This is a form of the TCB that is optimized for the gateway function.

   The PCI gateway writes a pointer in the GTCB to the RRIN register in the adapter.
4. The adapter side of the PCI gateway fetches the GTCB by DMA and interrupts the adapters processor. The gateway then creates a **transaction control block (TCB)** in the adapter address space. A TCB is a subset of an MCB. Finally the gateway calls the adapter IPN kernel to submit the TCB.
5. IPN calls the RAID-5 service for the addressed resource with a pointer to the TCB.
6. The RAID-5 service generates IPN transactions for each of the disk drives and sends these transactions to the disk service using the IPN kernel.
7. The disk service generates the appropriate SCSI read commands and passes them to the SSA driver.
8. The SSA driver issues the SCSI commands to the disk drives using the SSA protocol.
9. When the disk drives offer the requested data, the SSA driver transfers the data to SDRAM by DMA. The data is transferred to host memory through the PCI gateway.
10. When the drive returns good-completion status, the disk service calls IPN with the result of each transaction generated by the RAID-5 service.
11. When all the transactions have completed, the RAID-5 service calls IPN with the result of the original TCB.
12. IPN calls the adapter side of the PCI gateway. The gateway instructs the adapter hardware to DMA a pointer to the GTCB into the host request/reply queue. This queue is located in host memory. The pointer in the queue is tagged to indicate that the GTCB has completed. The PCI gateway then instructs the hardware to interrupt the host processor.
13. The host side of the PCI gateway fetches the pointer and calls IPN.
14. IPN calls the master process with the result of the original MCB.
15. The device driver will not return successful completion to the I/O request it received from the host operating system unless the adapter returned successful completion in the MCB from the device driver.

If a command issued by the adapter to the disk does not complete after all error recovery has been performed by the adapter, the transaction that caused this command is failed by a result code other than AS_Success. If an error is detected internally in the adapter while executing a transaction, the adapter stops further execution, sets an adapter error that describes the failure and activates a PCI interrupt. The transactions in progress at the time of the error are not completed. The device driver will then reset the adapter before resubmitting the unfinished transaction. An error detected on the PCI bus results in a Target Abort or Master Abort and the transaction is not completed.

If the transaction that has failed or not completed is to an array, the state of the array may change as a result of the error but its state will never be inconsistent with the condition of the data in the array.

Chapter 2. System-to-Adapter Interface

IPN Transactions

The adapter supports two different protocols for communication with the host software over the PCI:

- The IPN Transaction protocol is used in normal operation by the device driver to access the attached resources. Occasionally the adapter issues a transaction to the device driver to log an error. The transaction protocol is designed for performance and generality. It provides a simple (unordered) queuing model.
- The Command protocol is used for initialization, down-loading code, BIOS calls (PC Servers only) and Open Firmware (pSeries, RS/6000, or SP/2 systems only). This is a low-level protocol with restricted functions. Only one command can be issued to the adapter at a time.

Gateway Transaction Control Block (GTCB)

Each transaction passed over the PCI interface between the device driver and the adapter is described by a GTCB. It is built by the master side of the gateway in its local memory. The GTCB has a fixed length of 128 bytes and it must be aligned on a 16-byte boundary in the host PCI memory:

- The device driver normally issues a transaction to the adapter by writing a request containing the PCI memory address of the GTCB into the Request/Reply In (RRIN) register. The adapter fetches the GTCB from host PCI memory by DMA. When the transaction is completed the adapter enters a Reply_handle into the Request/Reply Queue (RR Queue).
- The adapter issues a transaction to the device driver by writing a request containing the PCI address of the GTCB into the RR Queue. The GTCB is in a host memory buffer previously allocated to the adapter by the device driver. The device driver accesses the GTCB using Load instructions. When the transaction is completed the device driver writes a request containing the PCI address of an OT_Done Slave Operation into the RRIN register.

The GTCB should remain allocated from the time the transaction request is issued to the master side of the gateway until the gateway returns the reply for the transaction and any associated abort.

Table 3. Format of a GTCB

Byte	3	2	1	0
0	Destination_node			
4	Destination_service			
8	Reserved = 00h	Major_function	Minor_function	
12 through 24	Parameter_DDR			
28 through 40	Transmit_DDR			
44 through 56	Receive_DDR			
60 through 72	Status_DDR			
76	Result_pointer			
80 through 108	Parameters			
112 through 120	Reserved			
124	Reply_handle			

Destination_node

This field contains a 32-bit unsigned integer to identify the IPN destination node. (The device driver uses the Initialize command to assign each adapter a unique number based on the host system and the PCI slot number occupied by that card.)

Destination_service

This field contains a 32-bit unsigned integer to identify the destination service.

- The registry has a fixed service number of 0000 0001h.
- The adapter service has a fixed service number of 0000 0007h.
- The service numbers for the disk service and the advanced function filters are dynamically allocated and may be obtained from the registry.
- The error logger connects itself to the registry during initialization.

Major_function This byte is coded as follows:

01h	**System**. All services must support the system transactions defined in “System Transactions” on page 301.
02h	**Application**. These transactions are defined separately by each service. See “IPN Storage Access Language (ISAL) Services” on page 159 for the transactions supported by the disk service and advanced function filters, “Registry Service” on page 131 for the transactions supported by the registry service, “Adapter Service” on page 204 for the transactions supported by the adapter service, and “Array-Configuration Service” on page 227 for the transactions supported by the configuration-agent service.

All other values are reserved.

Minor_function These 2 bytes select a particular system or application transaction.

Parameter_DDR
This 16-byte field contains the data descriptor for the transaction parameters. See “Data Descriptor (DDR)” on page 16 for the format of a data descriptor. The parameters are defined separately for each particular transaction.

Transmit_DDR This 16-byte field contains the data descriptor for data to be transmitted from the master to the slave.

Receive_DDR This 16-byte field contains the data descriptor for data to be received by the master from the slave.

Status_DDR This 16-byte field contains the data descriptor for the transaction status. The status, if any, is defined by the particular transaction.

Result_pointer This field points to the result word for the transaction. (See “Result Word” on page 18.)

- For transactions issued by the device driver, Result_pointer must contain a host PCI memory address; the adapter accesses the result word by DMA.
- For transactions issued by the adapter the result word is transferred by an OT_Done slave operation and the Result_pointer is not used.

Parameters This 32–byte field is available for inline parameters. Inline parameters are embedded within the GTCB starting at byte 80. Scatter/Gather is not supported.

If the length of the parameters is less than or equal to 16 bytes then the parameters must be inline and the type, address and offset fields in the parameter DDR are undefined. If a transaction requires from 17 to 32 bytes of parameters then they may be inline, in which case the DDR type must be DT_Inline, the offset must be zero and the address is undefined.

Transactions issued by the device driver: If the DDR length is greater than 16 bytes and the type is not DT_Inline then the adapter will perform an additional DMA operation to fetch the parameters from PCI memory.

Transactions issued by the adapter: If the DDR length is greater than 16 bytes and the type is not DT_Inline, the device driver must issue a slave operation to get the parameters.

Reply_handle This 4-byte field is returned to the device driver in the RR queue when the transaction completes.

For transactions issued by the adapter the Reply_handle is used in slave operations to identify the GTCB.

The Reply_handle is assigned by the master and it may have any convenient value except that the 4 low order bits must be 0000b. For example, the device driver could use the virtual address of the GTCB as the Reply_handle.

Data Descriptor (DDR)

A DDR is a component of the GTCB or Slave operation that provides the parameters, the receive data area, the transmit data, or the status area for a GTCB or slave operation. It always refers to PCI memory that resides in the host.

Table 4. Format of a Data Descriptor

Byte	3	2	1	0
0	Type	SG_length	Parameter	
4	Address			
8	Offset			
12	Data_length			

Type This field is coded to select one of the following types:

DT_Null	No data is present.
DT_Pci	The address field contains the PCI memory address of the data.
DT_PciScatGat	This DDR operates as DT_Pci with the addition of scatter/gather. The address field contains the PCI memory address of a scatter/gather list whose entries point to the data.

DT_Inline	The parameters are embedded in the parameter field of the GTCB and the address field is not used. This DDR must only be used for parameters and the parameter length must be from 17 to 32 bytes inclusive.

All other values are reserved. Note that the adapter may specify a reserved DDR type when issuing a transaction to the host. This indicates that the host must perform a slave operation to access the data.

SG_length
This 1-byte field is ignored by the Advanced SerialRAID Adapter. It was used in previous adapters to specify the number of entries in the scatter/gather list.

Parameter
This field is reserved.

Address
This 4-byte field contains the PCI address of the data or a scatter/gather list.

Offset
This unsigned integer contains an offset to be added to the base address of the data buffer to locate the first byte of the data to be transferred. When a host data buffer is addressed by a scatter/gather list, the Advanced SerialRAID Adapter searches the list to find the correct starting entry.

If Type = DT_Inline, then the Offset field must be set to zero.

Data_length
This unsigned integer specifies the length of the buffer in bytes that is available for data transfer. The scatter/gather list must have enough entries to locate this number of bytes after applying the offset.

Scatter/Gather List

The scatter/gather list is a variable-length list that allows data to be relocated in a virtual-memory environment. It has the same format as an OS/2 scatter/gather list. The list entries describe the data fragments in turn. Each entry specifies the PCI memory address and length of a fragment.

Table 5. Format of a scatter/gather list

Byte	3	2	1	0
0	Address_1			
4	Reserved=00h	Length 1		
8	Address 2			
...	...			
8n–4	Reserved=00h	Length_n		

Address	This field contains the PCI address of a fragment of data. The address may be on any byte boundary.
Length	This field contains an unsigned integer that is the length of the fragment in bytes. The length may be any number of bytes from 0 to $(2^{24}) - 1$.

Result Word

The result word is used to return the results of a transaction:

- For transactions issued by the device driver it must be aligned on a 4-byte boundary in host PCI memory
- For transactions issued by the adapter the Result word is included in the parameter block for the OT_Done slave operation.

Table 6. Format of a Result word

Byte	3	2	1	0
0	Reserved = 00h	Network_result	Application_result	

Network_result	This field is provided to report errors in the IPN network, for example an illegal destination service or when a transaction is routed through the peer gateway to another adapter. For transactions issued by the device driver, this field should be preformated with 00h, indicating no error. This avoids the need for the adapter to update it when there is no error.
Application_result	This field contains errors reported by the destination service. The device driver should preformat this field with 0000h, indicating successful completion. The specific errors reported are defined in “Application Results” on page 302.

RRIN register

This is a 4-byte register which the device driver uses to issue a request or reply to the adapter.

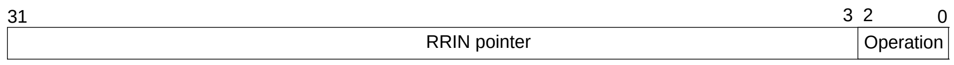


Figure 4. RRIN Register

Bits	**Name**	**Description**
31–3	RRINptr	**RRIN Pointer**. This is a pointer to a GTCB or the parameter block for a command as defined below.
2–0	RRINop	**RRIN operation**. The device driver sets this 3-bit field as follows: **000b** **Transaction Request.** This indicates that the device driver is issuing a transaction to the adapter. Bits 31–3 contain the PCI memory address of the GTCB. **001b** **Other Request.** This indicates that the host software issuing a slave operation or a command. Bits 31–3 contain the PCI memory address of the parameter block. **010b** Reserved **011b** Reserved **100b** Reserved **101b** Reserved **110b** Reserved **111b** Reserved If RRINop = x0xb then the adapter fetches the GTCB or the parameter block into adapter memory by DMA.

Request/Reply queue (RR queue)

The RR queue is used by the PCI gateway as a communication pipe from the adapter to the device driver. The pipe is a circular list in PCI memory that the adapter accesses with DMA. Each queue element is 4 bytes long and is shown in Figure 5.

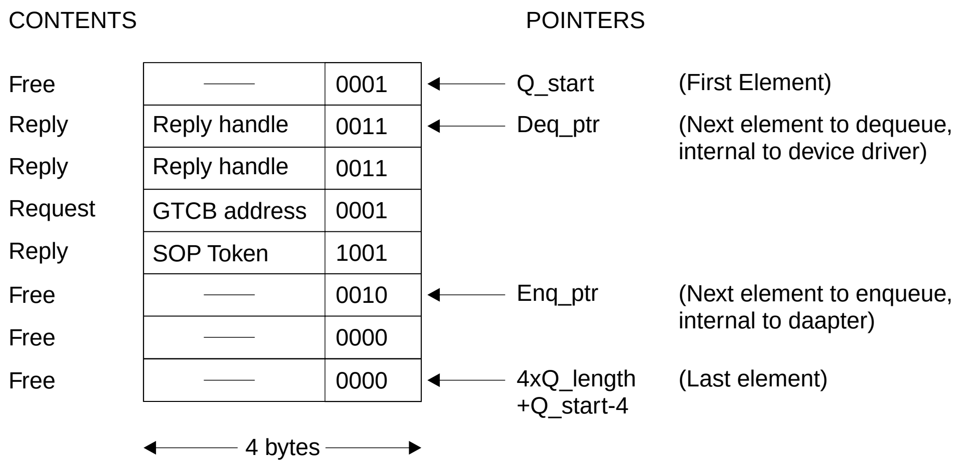


Figure 5. Request/Reply (RR) Queue

Bits 3 to 0 of each element are used for identification as follows:

0000b Transaction request from the adapter (Phase 0). Bits 31 to 4 contain the PCI memory address of the GTCB.

0001b Transaction request from the adapter (Phase 1). Bits 31 to 4 contain the PCI memory address of the GTCB.

0010b Reply to a transaction issued by the device driver (Phase 0). Bits 31 to 4 contain the GTCB Reply_handle.

0011b Reply to a transaction issued by the device driver (Phase 1). Bits 31 to 4 contain the GTCB Reply_handle.

0100b Reserved.

0101b Reserved.

0110b Reserved.

0111b Reserved.

1000b Reply to a successful slave operation issued by the device driver (Phase 0). Bits 31 to 4 contain the token from the slave operation.

1001b Reply to a successful slave operation issued by the device driver (Phase 1). Bits 31 to 4 contain the token from the slave operation.

1010b Reply to a unsuccessful slave operation issued by the device driver (Phase 0). Bits 31 to 4 contain the token from the slave operation.

1011b	Reply to a unsuccessful slave operation issued by the device driver (Phase 1). Bits 31 to 4 contain the token from the slave operation.
1100b	Reserved.
1101b	Reserved.
1110b	Reserved.
1111b	Reserved.

Bit 0 is a phase flag which prevents the device driver (dequeue agent) from fetching elements not yet stored by the adapter (enqueue agent).

Before issuing any transactions, the device driver must issue an Initialize command. This specifies the start address of the RR queue (Q_start), the total number of words in the queue (Q_length), and the maximum number of outstanding requests that the device driver is allowed (DD_max_requests).

The adapter must always leave at least one free element in the RR queue plus sufficient elements for replies to the maximum number of outstanding requests from the device driver, A_max_requests.

For the Advanced SerialRAID Adapter:

1 ≤ DD_max_requests ≤ 512

1 ≤ A_max_requests ≤ 26

DD_max_requests + (A_max_requests + 1)≤ Q_length ≤ 544

Initially the device driver fills the RR queue with dummy phase 1 request elements and sets its local variables as follows:

Deq_ptr = Q_start	Set dequeue pointer to start of the RR queue
Deq_phase = 0	Expected phase of dequeue elements.
DD_requests = DD_max_requests	Available number of outstanding requests.

Initially the adapter sets its local variables as follows:

Enq_ptr = Q_start	Set enqueue pointer to start of the RR queue
Enq_phase = 0	Phase of enqueue elements.
A_requests = A_max_requests	Available number of outstanding requests.

The adapter proceeds as follows to enqueue an element:

1. If enqueuing a request, check that A_requests > 0.
2. If enqueuing a transaction request, build the GTCB in local address space.
3. If enqueuing a transaction reply with a non-zero result, store the result word.
4. Store the element atomically at Enq_ptr, setting the three low order bits as previously defined.
5. Interrupt the device driver by setting RRQval to 1b in the PCI interrupt register.

6. Advance Enq_ptr to the next free element. If Enq_ptr wraps around to the beginning of the queue, toggle Enq_phase.
7. If enqueuing a request, decrement A_requests.

When the device driver is interrupted it repeats the following procedure until the pipe is empty:

1. Fetch the element addressed by Deq_ptr. If the phase flag does not match Deq_phase the pipe is empty.
2. Otherwise if the element is a reply, increment DD_requests.
3. Advance Deq_ptr to the next element. If Deq_ptr wraps around to the beginning of the pipe, toggle Deq_phase.
4. Process the element dequeued.

This protocol ensures that:

1. The adapter will always have space in the RR queue to reply to all outstanding transactions from the device driver. Hence elements will not be over-written by the adapter before they have been dequeued by the device driver.
2. There are no dynamic shared variables to control the RR queue.

Slave Operations

The device driver uses one or more slave operations to process a transaction issued by the adapter to the device driver. A slave operation requests the adapter to transfer the transaction parameters, data, status, or results to or from the host PCI memory by DMA. This avoids the need for the device driver to issue a large number of programmed I/Os to access this information in adapter memory.

The device driver issues a slave operation by writing the PCI memory address of a parameter block into the RRIN register and setting the low-order bits as previously described. The adapter fetches the parameter block by DMA and executes the specified operation. The adapter informs the host when the slave operation is complete by writing a token into the RRQ and raising a PCI interrupt in the usual way. Bit 1 of the token indicates whether the operation was successful.

Table 7. Format of a Slave Operation Parameter Block

Byte	3	2	1	0
0	Reserved = 00 0000h			Operation_code
4	Reply_handle			
8	Offset			
12–24	Data Descriptor			
28	Length_pointer			
32	Result_word			
36	Token			
40	Error_pointer			
44–124	Unused			

Operation_code
This defines the slave operation to be performed. The following codes are defined:

OT_Parms	Transfer the parameters to host PCI memory.
OT_Fetch	Transfer the transmit data to host PCI memory.
OT_Store	Transfer the receive data to the adapter.
OT_Status	Transfer the status to the adapter.
OT_Done	Transfer the status and result word and return the transaction buffer to the adapter.
OT_FastDone	Transfer the result word only and return the transaction buffer to the adapter.

Reply_handle This field identifies the GTCB for this slave operation. It is copied from the parent GTCB.

Offset This unsigned integer is added to the starting address of the data to locate the first byte to be transferred. Consequently the transfer can start at any point in the adapter data buffer.

Data_descriptor
A 16-byte field that defines the host PCI memory address and length of the information to be transferred. The information is defined in "Data Descriptor (DDR)" on page 16.

Length_pointer A pointer to a 4-byte word in host PCI memory in which the adapter stores the actual length of the data transferred by the operation.

Result_word This field contains the result of the transaction. It is only valid for the OT_Done and OT_FastDone operations.

Token A 4-byte field that is assigned by the device driver to identify this particular slave operation. It can have any convenient value except that the 4 low-order bits must be 0000b. The adapter returns the token in the RR queue when the slave operation has been completed.

Error_pointer The device driver can optionally set this field to point to a word in host PCI memory. Then if the slave operation fails the adapter will store 4 bytes at the specified address to identify the error. The first byte contains the IPN error code (DE_xxxx) and the remaining bytes are set to 00 0000h.

The 2 least-significant bits of Error_pointer must be 00b. If Error_pointer = 0000 0000h (Null) then no error code is stored.

GTCB Processing

The following steps describe the life cycle of a GTCB issued by the device driver and the control structures involved:

1. The device driver builds the GTCB on an 8-byte boundary in PCI memory.
2. If required, the device driver builds scatter/gather lists for the parameters, data, and status.

3. The device driver initializes the Result Word to 0000 0000h, indicating no error.
4. The device driver writes the PCI memory address of the GTCB to the RRIN register, ensuring that the 3 low-order bits are 000b.
5. The adapter hardware automatically moves the GTCB address to the RRIN Queue in adapter memory.
6. The adapter hardware removes the address of the empty buffer from the Free Queue and copies the GTCB into the buffer by DMA.
7. The adapter hardware adds the address of the buffer to the Receive Queue.
8. The adapter hardware interrupts the IPN microcode to process the GTCB.
9. If scatter/gather is specified, the adapter hardware fetches the elements from PCI memory by DMA, walking the list if the offset is non-zero.
10. The adapter hardware moves the data between the adapter buffer and PCI memory by DMA.
11. If there has been an error the adapter microcode initiates a DMA operation to store the result word in PCI memory.
12. The microcode enqueues an entry in the Send Queue containing the Reply_handle from the GTCB and the PCI memory address of the head of the RR Queue.
13. The adapter hardware automatically dequeues the Reply_handle and stores it in the RR Queue by DMA.
14. The adapter hardware sets RRQval to 1b in the PCI Interrupt Register. If it is not masked in the PCI Mask Interrupt register, this generates a PCI interrupt.
15. The device driver receives the interrupt, dequeues the Reply_handle from the RR Queue, resets the RRQval bit and then checks the RR Queue again.

A transaction issued by the adapter to the device driver is processed as follows:

1. During initialization the device driver issues an Initialize command to allocate one or more buffers to the adapter.
2. When the adapter needs to issue a transaction to the device driver it dequeues the address of the next free buffer and stores the GTCB by DMA.

   If the parameter length is 0 to 16 bytes, the adapter always stores the parameters within the GTCB. If the parameter length is 17 to 32 bytes the adapter *may* store the parameters in the GTCB. If so, it sets the DDR Type to DT_Inline. Otherwise the device driver must issue a slave operation to retrieve the parameters as described below.
3. The adapter issues the GTCB by writing its PCI memory address into the RR Queue and setting the 4 low-order bits as previously defined.
4. The adapter sets the RRQval bit in the PCI Interrupt register.
5. The device driver receives the PCI interrupt and dequeues the GTCB address from the RR Queue.
6. If required, the device driver issues an OT_Parms Slave Operation to obtain the parameters.
7. If required, the device driver issues one or more OT_Fetch or OT_Store Slave Operations to transfer the data.

8. If required, the device driver issues an OT_Status Slave Operation to return the transaction status.
9. The device driver issues an OT_Done or OT_FastDone Slave Operation which returns the transaction Result and it implicitly frees the transaction buffer for use in a subsequent transaction.

For each Slave Operation the adapter transfers the requested information to or from host PCI memory by DMA. This eliminates the need for the device driver to move the data using programmed I/O.

Timeouts

The adapter times a GTCB from arrival to reply. If the reply is not sent within two minutes and the transaction is not FN_REGY_TestResrcReady, the adapter sets the error register with the error code SS_TIMEOUT, interrupts the host, and waits to be reset. While waiting to be reset, the adapter does not respond to heartbeats or accept any transactions from the host.

Commands

The adapter command set provides a low-level interface to the adapter, for example for initialization, down-loading microcode, and BIOS or Open Firmware calls. Only one command can be in progress at a time. If the host has previously sent a command to the adapter, it should not send another command until it has received an interrupt from the adapter to indicate completion of the current command.

The command interface uses the following protocol:

1. The host software builds a 128-byte parameter block in host PCI memory.
2. The host writes a command request element containing the PCI address of the parameter block to the RRIN register.
3. The adapter fetches the parameter block by DMA and then interrupts the IPN microcode to process the request.
4. When the adapter completes a command successfully it sets ComDone to 1b in the PCI interrupt register.

   If a command fails the adapter sets either ComErr or CatErr to 1b in the PCI interrupt register. The adapter also stores an error code in the adapter error register.
5. The host software accesses the PCI interrupt register to determine the reason for the interrupt and to clear it.

Initialize

The Initialize command configures the RR queue which is used for IPN transactions and it allocates an IPN node number to the adapter.

Table 8. Parameter block for Initialize

Byte	3	2	1	0
0	Reserved = 00h	Environment	Reserved = 00h	Command = 30h
4	Q_start			00b
8	Q_length		DD_max_requests	
12	Node			
16	Reserved = 0000 0000h			
20 through 124	Buffer_list			

Q_start This field contains the PCI memory address of the first word in the RR queue. It must be aligned on a 4–byte boundary. The queue must be allocated in contiguous host memory without using scatter/gather.

Q_length This 2-byte unsigned integer specifies the number of elements allocated to the RR queue.

DD_max_requests + A_max_requests + 1 ≤ Q_length ≤ 544

DD_max_requests

An unsigned integer specifying the maximum number of outstanding requests that the device driver is allowed to originate.

1 ≤ DD_max_requests ≤ 512, otherwise the adapter reports an exception for Invalid Parameter.

Node This field contains the IPN node number assigned to the adapter by the device driver.

Buffer_list This field contains a list of 1 to 26 buffers that the device driver is allocating to the adapter for use in subsequent transactions initiated by the adapter. Each entry in the list is the PCI address of a buffer in host memory. The last entry must be null (0000 0000h). Each buffer must be aligned on a 16-byte boundary and have a length of 128 bytes.

The device driver can only reclaim the buffers by resetting the adapter.

The following exceptions may be indicated in the PCI Interrupt register:

- Catastrophic error. Further details of the error are provided in the Error Type field in the Adapter Error register, that is, SS_WRONG_INT or SS_WRONG_INI_PARMS.

After an Initialize Command only the Execute I/O command may be issued before the next adapter reset.

Download

The Download command allows updated microcode to be downloaded into the flash memory on the adapter. The microcode load includes the BIOS code or Open Firmware (depending on the host system), POSTs, and functional code. The Download command must be immediately preceded by a software reset.

After the Download command has been executed the adapter must be reset again before any further commands or transactions can be issued.

Table 9. Parameter block for Download

Byte	3		2	1	0
0	Reserved				Command = 31h
4	G	Reserved = 000 0000b	Reserved = 00h	Unused	
8	Address				
12	Length				
16	LRC				
20 through 24	ROS Level				
28 through 124	Unused				

Gather (G)	If byte 7, bit 7 is set to 1b then the Address parameter is the host PCI memory address of a scatter/gather list. Otherwise Address is the PCI memory address of the microcode itself.
Address	This field contains the host PCI memory address of the microcode or a scatter/gather list which locates the microcode. Addresses are aligned on 4-byte boundaries.
Length	This field contains a 32-bit unsigned integer which specifies the length of the microcode in bytes. Length is a multiple of 4.
LRC	This word contains a Longitudinal Redundancy Check (LRC) to ensure integrity of the microcode. The LRC is formed by adding each word of the microcode to the constant AAAA AAAAh, using 32-bit arithmetic.
ROS level	This 8-digit ASCII-coded field contains the level of the flash EPROM after the download. This is the value that is reported in the 8 most significant bytes of the RL field of the VPD; the least significant 4 bytes of that field are zero.

The updated microcode is downloaded as follows:

1. The host software resets the adapter by setting SoftRst to 1b in the BIST control register
2. The adapter disables the SSA ports and executes the boot sector of the flash memory only.
3. The host issues the Download command.

4. The adapter fetches the microcode to SDRAM by RMA. If the LRC is good, the adapter writes the new microcode to flash EPROM. The adapter generates a host interrupt by setting ComDone to 1b in the PCI Interrupt register to inform the host that the command has completed.
5. The host resets the adapter again.
6. The host software must issue a command other than Download before the adapter will leave the boot sector and start the functional code.

The following exceptions may be indicated in the PCI interrupt register:

- Catastrophic error. Further details of the error are provided in the adapter error register.

Execute I/O

The Execute I/O command provides a simple synchronous I/O interface to support system IPL and software installation. It should normally be used only by the BIOS or Open Firmware (as relevant to the host system), not by the device driver.

Execute I/O can perform only one I/O operation at a time.

Table 10. Parameter Block for Execute I/O

Byte	3	2			1	0
0	Operation	M	P	Reserved = 000000b	Reserved = 00h	Command = 32h
4	Disk					
8	LBA					
12	Length					
16	Buffer_address					
20	Reserved = 000000h					Status
24 through 124	Unused					

Operation This byte is coded as follows to specify the function to be performed:

01h **Inquiry.** This operation checks that the disk is ready. If it is, the following descriptor (up to 32 bytes) is stored in host memory at the address in the Buffer-address field.

Block_size A 4-byte unsigned integer specifying the block size in bytes.

Capacity A 4-byte unsigned integer specifying the disk capacity in blocks.

Serial_number 16 bytes containing the ASCII serial number of the resource.

Resource ID A 4-byte unsigned integer identifying the resource.

	02h	**Ready Test.** The command completes successfully when all the attached resources are ready or when the time period in seconds defined in the length field has expired, whichever is the shortest time. The value of the physical (P) bit determines if these are logical or physical resources. If the mode bit (M) is 0b, the logical resources are of owning-module type DriverManualDisk; if the mode bit is 1b, they are of type DriverAutomaticDisk. A 4-byte unsigned integer that specifies the number of attached resources that are ready is stored in host memory at the address provided in the Buffer-address field. The list of resources is retained in the adapter.
	03h	**Start Transaction.** This operation is used to read and write by the command protocol. This is used by the BIOS for PC servers only. The type of DDR must be either DT_Pci, DT_PciScatGat, or DT_Null.
	05h	**Read NVRAM.** The byte in adapter NVRAM addresses by the LBA field is stored in host memory at the PCI address in the Buffer_address field.
	06h	**Write NVRAM.** The data byte in host memory at the PCI address in the Buffer_address field is stored in adapter NVRAM at the address in the LBA field. The Read and Write NVRAM functions are used by the PC BIOS in PC server systems.
	10h	**Read.** This operation reads the data blocks identified by the Disk, LBA, and Length parameters
	11h	**Write.** This operation writes the data blocks identified by the Disk, LBA, and Length parameters
Mode (M)	The mode bit controls the interpretation of the Disk field and the type of resources listed to a Ready Test operation.	
	0b	Disk is a Resource_ID. (This mode is used by the CBIOS interface routine on PC systems as a Ready Test operation.) If Mode = 0b, the Ready Test operation creates a list of all resources having an owning module type of DriverManualDisk that are controlled by the disk services or by a filter.
	1b	Disk is an index (zero origin) into a list of configured disks created by the last Ready Test operation. (This mode is used to boot the system using Open Firmware or by the BIOS on PC systems for reads and writes.) If Mode = 1b, the Ready Test operation creates a list of all resources having an Owning Module Type of DriverAutomaticDisk.

Physical (P)	The bit is only used during the Ready Test operation to qualify the Disk field: **0b** — The Disk field contains a Resource_ID and Ready Test refers to logical resources. The Physical bit should be 0b for normal IPL operations to ensure that the IPL device is a logical resource. **1b** — The Disk field contains a Resource_ID and Ready Test refers to physical resources. The physical field should be set to 1b to obtain the Serial Number of each physical resource using Ready Test and Inquiry operations. This can be executed after an unsuccessful completion of the Diagnostic operation to compare the Serial Numbers of good physical resources with those reported after a successful IPL process. If the disks attached to the adapter are configured into arrays, the number of logical resources may not be the same as the number of physical resources.
Disk	An unsigned integer to select a particular resource according to the specified Mode field. (See the definition of the Mode field for more details.)
LBA	An unsigned integer specifying the starting logical block address for a read or write operation. It is also used for Read/Write NVRAM operations to define the offset in NVRAM for the byte to be read or written.
Length	An unsigned integer specifying the number of blocks to be accessed in a read or write operation. It is assumed that the host memory buffer is large enough for the read data. When the Ready Test operation is specified, the Length field defines the number of seconds allowed for all the resources to become ready.
Buffer_address	The PCI memory address of a buffer in host memory for read/write data or IPN directive.
Status	This byte is reserved.

The following conditions may be reported in the Interrupt register with a reason code in the adapter error register.

- Catastrophic error, for example an invalid parameter.
- Device or attachment error. The system may be able to recover by using an alternative device.

Diagnostic Area

The Diagnostic Area provides access to the VPD and the adapter dump. It is stored at a fixed offset of 003F FEC0h from BAR_1. All fields are written by the Advanced SerialRAID Adapter and read by the host.

Table 11. Diagnostic Area

Byte	3	2	1	0
0 through 40	Cache Vital Product data			
44 through 60	Not defined			
64 through 224	Adapter Vital Product Data			
228 through 272	Not defined			
276	Trace Point			
280 through 284	Disk Unique ID			
288	Dump Start LBA			
292	Sequence		Reserved = 00h	Dump status
296 through 316	Not defined			

Cache VPD This field contains 40 bytes of VPD plus a 4–byte checksum.

Adapter VPD This field contains 161 bytes of VPD plus 3 pad bytes containing 00 0000h. Refer to “Vital Product Data” on page 33 for details about the contents of the VPD.

Trace Point This field contains a trace pointer when Error Code = SS_SENSE.

Disk Unique ID The SSA Unique ID of a disk drive where the adapter has written the dump following a showstop error.

Dump Start LBA
The location of the dump on disk.

Dump Status This byte indicates the presence of a dump on the hot spare disk. (A dump to disk will only be taken if a hot spare is available.)

Sequence This unsigned integer contains a version number from NVRAM. It is incremented for each dump.

Resets

The actions taken for the various possible resets of the adapter are defined in this section. In the table below, the term *Node* refers to the adapter node in the SSA loop; there are two nodes on the adapter.

Table 12. Advanced SerialRAID Adapter Reset Actions

	PCI Reset	**Software Reset**	**Start BIST**	**Total reset**	**Absolute Reset (Note 1)**	**Link Reset**
Reset PCI Configuration registers	Yes	No	Yes	No	No	No
Reset chips on the adapter	Yes	Yes	Yes	No	Yes	No
Run BIST	No	No	Yes	No	No	No
Run POST and SDRAM check-out	Yes	No	Yes	No	No	No
Reset SSA Ports	2 Nodes (H/W)	2 Nodes (H/W)	2 Nodes (H/W)	1 Node (F/W)	2 Nodes (H/W)	No
Wrap and test SSA Ports	2 Nodes	2 Nodes	2 Nodes	No	2 Nodes	No
Clear SSA Configuration table and IPN Registry	2 Nodes	2 Nodes	2 Nodes	1 Node	2 Nodes	No
Internally purge SSA commands	2 Nodes	2 Nodes	2 Nodes	1 Node (Note 2)	2 Nodes	No
Async_alert (Note 3)	Yes	Yes	Yes	Yes	Yes	No
Re-configure SSA network (Note 4)	Yes (Note 5)	Yes	Yes	Yes	Yes	No

Notes:

1. On receipt of an absolute reset the firmware initially stops execution, sets a showstop error code and interrupts the host. If the device driver does not issue a software reset within a certain time-out, the adapter automatically takes the actions shown in Table 12 for a software reset.
2. The SSA commands that were purged are reissued after the SSA network has been reconfigured.
3. The adjacent SSA Node sends an Async_alert (’Remote Port Disabled’) when the adapter Wraps or Disables a port. The master initiator should then send a Master_alert to every other initiator to remove that link from its configuration table.

   The adjacent SSA Node sends another Async_alert (’Port now operational’) when the adapter enables one of its ports and it becomes ready. The master initiator should then send a Master_alert to all other initiators to add the link into their configuration tables.
4. When a port becomes ready after a reset, the adapter proceeds as follows:
   - Walk the network and build the configuration table by issuing a Query_node message to each SSA node accessible through that port.

- Issue a quiesce message to each SCSI target to purge all commands from this adapter and remove any stale Return_paths from the initiator tables.
- Issue another Query_node message to each SCSI target to register a new Return_path in the initiator table.
- If the Query_node_reply messages indicate that the Advanced SerialRAID Adapter should be master, it will issue a Configure_port message specifying 'Set normal mode' to each port that is operational. The adapter will then send a Master_alert message specifying 'All ports now operational' to each primary initiator.

5. After PCI RST# is asserted and the resulting SSA reconfiguration, if the adapter is the only initiator in the network, it issues a Clear_queue message to each target before issuing any SCSI commands. This ensures that all commands outstanding from any previous initiator are purged.

Vital Product Data

Vital Product Data (VPD) is information that uniquely defines the adapter card. The device driver can retrieve the adapter VPD from the diagnostic area in SDRAM through BAR_1 (see “Diagnostic Area” on page 31 for more details). For compatibility with the systems host software this is similar to, but not identical to, the VPD format recommended in revision 2.1 of the PCI Local Bus specification.

The VPD fields supported are:

Part Number (PN)	This is the 8 alphameric character ASCII-coded part number of the adapter card. If fewer than eight characters are used it is right-justified and padded with zeros on the left.
FRU Part Number (FN)	This is the 8 alphameric ASCII-coded part number of the field-replaceable card unit. If fewer than eight characters are used it is right-justified and padded with zeros on the left.
Serial Number (SN)	This is an 8 alphameric character ASCII-coded FRU serial number. This serial number is unique for the FRU part number and is part of the manufacturing serial number printed on the card. The serial number is in the range 00000000 through ZZZZZZZZ.
Engineering Change Level (EC)	This is a 10 alphameric character ASCII-coded Engineering Change (EC) level number. This number is updated whenever a hardware or microcode change is made on the card. If fewer than ten digits are used, the leading digits are padded with zeros.
Manufacturing Location (MF)	This 6 alphameric character ASCII-coded field indicates the plant of manufacture.
ROS Level (RL)	This 12 alphameric character ASCII-coded field

	indicates the ROS level of the card. A value of 00000000 in this field indicates that the POST code has detected a check-sum error in the code and a new version of code must be downloaded before the adapter can become fully operational. The SSA Adapter Microcode diskette, which is shipped with each adapter card, contains a version of adapter microcode that recovers this error in the event of the host system being unable to IPL because of this failure.
Loadable Microcode Level (LL)	This 2-digit ASCII-coded field indicates the version of loadable microcode required for satisfactory operation of this card.
Device Driver Level (DD)	This 2-digit ASCII-coded field indicates the minimum level of device-driver program required for this level of card.
Description of Function (DS)	This ASCII-coded field describes the function of this adapter card. This is 'SSA-ADAPTER'.
SDRAM Size (Z0)	This ASCII-coded field contains the characters 'SDRAM=' followed by three characters indicating the size of the installed SDRAM in megabytes.
Fast-Write Cache Size (Z1)	This ASCII-coded field contains the characters 'CACHE=' followed by 3 characters indicating the size of the installed Fast-Write cache card in megabytes. If no cache is installed, the size character is '000'.
Unique ID (Z2)	This ASCII field begins with the characters 'UID=' followed by 16 ASCII characters that report the lowest of the two 8-byte hexadecimal SSA Unique IDs in ASCII format. The right-most character is an even number.

An example of the layout of the adapter card VPD is:

```
V P D (00) L X X
* P N (06) 1 2 3 4 5 6 7 8
* F N (06) 1 2 3 4 5 6 7 8
* S N (06) 1 2 3 4 5 6 7 8
* E C (07) 1 2 3 4 5 6 7 8 9 A
* M F (05) I B M 9 0 2
* R L (08) 0 3 0 1 0 0 0 0 0 0 0 0
* L L (03) 0 5
* D D (03) 0 0
* D S (08) S S A - A D A P T E R
* Z 0 (06) S D R A M = 0 6 4
* Z 1 (06) C A C H E = 3 2
* Z 2 (10) U I D = 0 0 0 0 1 2 3 4 5 6 7 8 9 A B C
```

The decimal number in ( ) is the inclusive descriptor length divided by 2 except for the first line. Each descriptor field including the first 4 identification characters must be an even length. Some fields, for example, the *DS field, may have to be padded with a null character to make it an even length.

L is the inclusive VPD field length divided by 2, starting at the eighth byte, that is the first *.

XX in the first line are reserved for the CRC value. These bytes are 0000h as the adapter does not implement the CRC.

An example of the VPD is:

```
Hex Address      Data
(Offset)

0000             56 50 44 00 4D 00 00
0007             2A 50 4E 06 31 32 33 34 35 36 37 38
0013             2A 46 4E 06 31 32 33 34 35 36 37 38
001F             2A 53 4E 06 31 32 33 34 35 36 37 38
002B             2A 45 43 07 31 32 33 34 35 36 37 38 39 41
0039             2A 4D 46 05 49 42 4D 39 30 32
0043             2A 52 4C 08 30 33 30 31 30 30 30 30 30 30 30 30
0053             2A 4C 4C 03 30 35
0059             2A 44 44 03 30 30
005F             2A 44 53 0A 53 53 41 2D 41 44 41 50 54 45 52 20
006F             2A 5A 30 07 53 44 52 41 4D 3D 30 36 34 20
007D             2A 5A 31 06 43 41 43 48 45 3D 33 32
0089             2A 5A 32 0C 55 49 44 3D 30 30 30 30 31 32 33 34 35 36 37 38 39 41 42 43
```

System Boot

The Advanced SerialRAID Adapter supports booting the host PC system from an SSA disk or array and booting a non-PC system from only a non-RAID disk if the adapter code level is lower than level 50, or from a non-RAID disk or array if the code level is 50 or higher.

Systems based on the Common Hardware Reference platform (CHRP) use Open Firmware. The FCODE routines are stored in the PCI Expansion ROM on the adapter card and interpreted by the host processor.

PC Servers use the Basic Input/Output sub-system (BIOS). The ‘x86’ code is stored in the PCI Expansion ROM on the Advanced SerialRAID Adapter and executed by an Intel-compatible host processor.

The execute I/O command is used to issue read/write operations to the adaptor.

The following sequence of Execute I/O commands may be used in the bootstrap code:

1. The host issues a Ready Test operation to determine how many resources are available.

2. The host issues a Test operation to each value of the disk field to find the boot resource with the required serial number.
3. Finally the host issues a Read operation to the corresponding disk to retrieve the boot record

Expansion ROM

The expansion ROM contains an Open Firmware image and a BIOS image. These images are included as part of the adapter microcode that is updated using the download command. The expansion ROM is stored in flash memory. It is accessed by the host processor through PCI memory space using the BAR_6 configuration register. The size of the expansion ROM is 16 KB.

There are 3 components in each ROM image:

1. The ROM header is located at the beginning of each image.
2. The PCI Data Structure is located by an offset in the ROM header.
3. Code. For PC servers, this contains Intel x86 code for BIOS. For Open Attach systems, this contains FCode.

ROM Header — PC Servers

Table 13 shows the contents of each ROM header in the BIOS image. The offset of each field is the number of bytes from the beginning of the image and the length is in bytes:

Table 13. Format of Expansion ROM in the BIOS image

Offset	Length	Value	Description
0–1h	2	55AAh	**ROM signature.** This 2-byte field contains 55h in the first byte and AAh in the second byte.
2h	1	xx	**Initialization size.** This 1-byte field identifies the size of the code in units of 512 bytes. (The size may be reduced by the INIT function.) The last byte of the code contains a checksum.
3–5h	3	xx	**INIT entry point.** The system BIOS does a Far CALL to this location.
6–17h	18	xx	Reserved.
18–19h	2	xxxx	**Pointer to PCI Data structure.** This is a 2-byte offset in little-endian format that points to the PCI data structure. The reference point for this offset is the beginning of the ROM image.

ROM Header — Open Firmware

Table 14 shows the contents of each ROM header in the Open Firmware image. The offset of each field is the number of bytes from the beginning of the image and the length is in bytes:

Table 14. Format of Expansion ROM in the Open firmware image

Offset	Length	Value	Description
0–1h	2	55AAh	**ROM signature.** This 2-byte field contains 55h in the first byte and AAh in the second byte.
2–3h	2	xxxx	**FCode pointer.** This is a 2-byte offset in little-endian format from the start of the ROM image to the FCode program.
4–17h	14	xx	Reserved.
18–19h	2	xxxx	**Pointer to PCI Data structure.** This is a 2-byte offset in little-endian format that points to the PCI data structure. The reference point for this offset is the beginning of the ROM image.

PCI Data Structure

Table 15 shows the contents of this structure. It is aligned on a 4–byte boundary. The offset of each field is the number of bytes from the beginning of the structure and the length is in bytes.

Table 15. Format of PCI Data Structure

Offset	Length	Value	Description
0–3h	4	‘PCIR’	**Signature.** These 4 bytes contain the string ‘PCIR’ with ‘P’ being at offset 0 and ‘R’ at offset 3.
4–5h	2	1014h	**Vendor Identification.** This 16-bit field has the same definition as the PCI Vendor ID register in the configuration space. The value assigned to IBM is 1014h.
6–7h	2	0058h	**Device Identification.** This 16-bit field has the same definition as the PCI Device ID register in the configuration space. The value assigned to the Advanced SerialRAID Adapter is 0058h.
8–9h	2	0000h	**Pointer to Vital Product Data.** This 16-bit field is an offset in little-endian format from the start of the ROM image to the Vital Product data (VPD). Since the Advanced SerialRAID Adapter VPD does not conform to revision 2.1 of the PCI specification, this field is set to 0000h.
A–Bh	2		**PCI Data Structure Length.** This 16-bit field defines the total length of the data structure, starting from the first byte in the Signature field. The length is in little-endian format and is in bytes.
Ch	1	00h	**PCI Data Structure Revision.** This 8-bit field identifies the data structure revision level. The revision level is 0.
D–Fh	3	00020Ch	**Class Code.** this 24-bit field has the same definition as the PCI Class Code register in the configuration space.

Table 15. Format of PCI Data Structure (continued)

10–11h	2	0008h	**Image Length.** This 16-bit integer defines the length of the image. The length is in little-endian format and the value is in units of 512 bytes.
12–13h	2	0	**Revision Level of Code/Data.** This 16-bit field defines the revision level of the code in the ROM image.
14h	1	0	**Code Type.** This 8-bit field identifies the type of code contained in the image: • 00h Intel x86 for a PC-AT compatible system • 01h Open Firmware for pSeries, RS/6000, or SP/2
15h	1		**Indicator.** Bit 7 of this field is 1b in the Open Firmware image to indicate that this is the last image in the expansion ROM. Bits 6–0 are always 000 0000b. (Reserved.)
16–17h	2	0000h	Reserved.

BIOS

The Advanced SerialRAID Adapter provides a subset of the compatibility BIOS for PC servers. This implements the INT 13h disk interface to allow DOS programs to communicate with the adapter. It also supports system IPL with INT 19h.

The BIOS is a simple, synchronous interface for executing one I/O operation at a time. It supports 8 disk drives only. In normal operation the device driver accesses the adapter directly without using any of the BIOS functions.

There are two software components involved in the implementation:

1. The majority of the function is implemented by internal adapter microcode that maps the request to IPN transactions. This microcode is invoked using the Execute I/O command (see “Execute I/O” on page 28). It is not a direct mapping of the standard BIOS register interface. In particular the disk is addresses by a Logical Block Address (LBA) rather than a Cylinder, Head, and Sector (CHS).
2. A small BIOS interface routine in ‘x86’ code implements the standard INT 13h and INT 19h calls. This routine copies the ‘x86’ registers to and from the Execute I/O Parameter block and translates the CHS address to and from an LBA. The interface routine also manages the fixed disk drive data area at address 40–74h and 40–77h in host memory.

   The BIOS interface routine is stored in the PCI Expansion ROM.

INT 13h Functions

Table 16 shows which INT 13h functions are supported by the ‘x86’ interface routine.

Table 16. INT 13h Functions

AH register	Functions and Restrictions
00h	Reset Disk System
01h	Read Status of Last Operation
02h	Read Desired Sectors into Memory

Table 16. INT 13h Functions (continued)

AH register	Functions and Restrictions
03h	Write Desired Sectors from Memory
04h	Verify Desired Sectors. (No action - Not implemented)
05h	Format Desired Cylinder (Invalid request - Not implemented)
06h	Format Desired Cylinder and Set Bad Sector Flags. (Invalid request - Not implemented)
07h	Format Drive starting at Desired Cylinder (Invalid request - Not implemented)
08h	Read Drive Parameters
09h	Initialize Drive Pair Characteristics (No action - Not implemented)
0A–0Bh	Reserved
0Ch	Seek. (No action - Not implemented)
0Dh	Alternate Disk Reset. (Executed as Reset Disk System)
0E–0Fh	Reserved
10h	Test Drive Ready
11h	Recalibrate. (No action - Not implemented)
12–14h	Reserved
15h	Read DASD Type. (Invalid request - Not implemented)
16–18h	Reserved
19h	Park Heads. (Invalid request - Not implemented)
1Ah	Format Unit. (The defect table and modifiers are not supported)
1B–40h	Reserved
41h	Check extensions present
42h	Extended read
43h	Extended write
44h	Extended verify (no action taken)
45–46h	Reserved
47h	Extended seek (no action taken)
48h	Extended get drive parameters
49–FFh	Reserved

Open Firmware

The Expansion ROM also provides an Open Firmware image that supports the following functions:

- Invoking the adapter Built-In Self-Test (BIST) at power-on.
- Booting the AIX operating system from a disk drive that is attached to the Advanced SerialRAID Adapter.
- Installing AIX onto SSA non-RAID disks.

The Open Firmware in the Advanced SerialRAID Adapter is intended for use with pSeries, RS/6000, or SP/2 systems that use the Common Hardware Reference Platform (CHRP) only.

Open Firmware is an industry standard (IEEE-1275) for an expansion ROM image that is independent of the particular instruction set supported by the host processor. The system configuration software probes the hardware and builds a hierarchical Device Tree that describes the physical configuration. Each node in the Device Tree represents a resource, for example a PCI bus, an adapter card, a disk drive, or an array. It is possible to specify a path through the Device Tree from the root and open a particular node, rather like selecting a working directory in a file system.

Each node has an associated Package that contains certain Properties, Methods, and private data. A Property is a descriptive item with a value and a name. A Method is a software procedure that performs a particular function such as reading a disk drive.

When the Advanced SerialRAID Adapter Open firmware image is probed it creates a parent Bus Node to represent the adapter card. The Bus node then creates a number of Child nodes, one for each IPN Logical Disk that is accessible. (Notice that there is no Node in the Device Tree to represent an SSA Port or an SSA Loop.)

The Device Tree also contains a special node, /packages, that is the parent of some standard support packages. For example, the deblocker support Package helps to implement a byte-oriented interface by using the block-oriented Methods of a disk drive and the disk-label support Package interprets system-dependent partitioning information.

An Open Firmware image consists of Fcode that is a tokenized version of Forth. The Fcode is interpreted by the host software to build the Device Tree and access the devices. Forth is a stack-based interpretative language using Reverse Polish notation. A Forth program consists of a sequence of words that operate on the stack and add new Words to the Dictionary. The unit of data on the stack is called a Cell. A Cell provides storage for at least 32 bits.

A Method is specified in terms of its effect on the stack using the following notation:

```
        (v w x - - y z)
```

In this particular case the Method removes the cells v, w, and x from the top of the stack, performs some calculation and returns cells y and z to the stack. In each list the right-most item is at the top of the stack.

The following sections define the contents of the Advanced SerialRAID Adapter Bus and Child Nodes that are provided by the expansion ROM itself. The system configuration software defines some additional Properties and Methods that are not listed here.

Bus Node

Table 17 shows the Properties created for the Bus node representing the Advanced SerialRAID Adapter.

Table 17. Bus Node Properties

Property	**Description**
name	A standard Property that specifies the generic name of the Device. All SSA adapters have the value ‘ssa’.
device_type	A standard Property that specifies the logical interface to the Device. SSA storage adapters have the value ‘ssa-scsi-2’.
compatible	A standard Property that is created by the system software. It contains a list of Devices that the current Device is compatible with, starting with the Device itself. For the Advanced SerialRAID Adapter the list contains: **pci1014,91** (The PCI Subsystem ID for the Advanced SerialRAID Adapter adapter. **pci1014,58** (The PCI Device ID for the adapter.) **pciclass,0c0200** (The PCI Class code for an SSA adapter.)
reg	A standard Property that describes the PCI configuration, I/O, and Memory spaces required by the Advanced SerialRAID Adapter adapter. Each address space descriptor consists of 3 cells that identify the PCI Base Address Register followed by 2 cells that specify the address range in bytes.
#address-cells	A standard Property that specifies the number of cells in the bus address of a Child Node. The value for the Advanced SerialRAID Adapter is 4.
#size-cells	A standard Property that specifies the number of cells in a bus address. The value for the Advanced SerialRAID Adapter is 0 since SSA in not a memory-mapped bus.
ssa-address-logical	This Property is present to indicate that a Child node address uses the logical format rather than the physical format, that is it is a 15-character IPN Serial number.

Table 18 shows the methods provided by the Bus Node.

Table 18. Bus Node Methods

Method	Syntax	Function
open	(– – okay?)	Open this adapter for use by setting the Bus Master, Memory Space, and I/O Space bits in the PCI Command register. (Here 'okay?' denotes a flag that is true if the operation was successful.)
close	(– –)	Close this (previously opened) adapter by clearing the Bus Master, Memory Space, and I/O Space bits in the PCI Command register.
decode-unit	(addr len – – phys.lo phys.mid1 phys.mid2 phys.hi)	Convert the textual representation of a Child address to numerics. (Open Firmware uses the numerical form to store an address on the stack, in a Property value, or as the argument to a method.)
encode-unit	(phys.lo phys.mid1 phys.mid2 phys.hi – –addr len)	Convert the numerical representation of a Child address to text, that is a 15-character IPN Serial Number in ASCII.

Child Nodes

Table 19 shows the Properties created for a child Node.

Table 19. Child Node Properties

Property	Description
name	All IPN Logical Disks have the value 'disk'.
device_type	All Logical disks have the value 'block'.
reg	A standard Property that is used here to define a unit address for the device. It consists of 4 Cells that contain 01h followed by the 15-character IPN Serial number for ASCII.
driver_type	The value of the mode and Physical parameters for the Ready Test operation of the Execute I/O command (80h).
resource_id	The IPN Resource ID of the device.
capacity	The device capacity in blocks.
block_size	The block-size of the device in bytes (512).

Table 20 shows the Methods provided by a Child Node.

Table 20. Child Node Methods

Method	Syntax	Function
open	(– – okay?)	Open the Logical Disk corresponding to the current Node for use.
close	(– –)	Close this (previously opened) Logical Disk.

Table 20. Child Node Methods (continued)

Method	Syntax	Function
load	(addr – – len)	Load a boot program from the current Logical Disk into memory at the specified address and return the program length in bytes. This method uses the disk-label support package.
read	(addr len – – actual)	Read the current Logical Disk into the specified memory buffer and return the actual number of bytes transferred. This Method uses the deblocker support package.
write	(addr len – – actual)	Write the current Logical Disk from the specified memory buffer and return the actual number of bytes transferred. This Method uses the deblocker support package.
seek	(pos.lo pos.hi – –status)	Set the Device position for the next read or write. Return 0 if the operation succeeds, else -1. This Method uses the deblocker support package.
offset-low	(– – u)	Return the lease-significant cell of the starting offset of the disk partition. This Method is defined in the device Support Extensions. The offset is obtained by calling the Offset Method in the disk-label support package with an argument of 0.
offset-high	(– – u)	Return the most-significant cell of the starting offset of the disk partition.
block-size	(– – block-len)	Return the block size of the Logical Disk in bytes.
max-transfer	(– – max-len)	Return the size of the largest possible transfer in bytes, rounded down to a multiple of the block size.
read-blocks	(addr block# #blocks – – #read)	Read #blocks beginning at block# into memory, starting at addr. Return the number of blocks actually transferred. This Method is used by the deblocker support package.
write-blocks	(addr block# #blocks – – #written)	Write #blocks beginning at block# from memory, starting at addr. Return the number of blocks actually transferred. This Method is used by the deblocker support package.
dma-alloc	(size – – virt)	Allocate size bytes of contiguous memory and return the virtual address.
dma-free	(virt size – –)	Free the memory previously allocated by dma-alloc.

The following codes are displayed for the Advanced SerialRAID Adapter when booting with open firmware:

Code	Description
E600	SSA PCI adapter open firmware has run successfully.
E601	BIST has been started but failed to complete after 4 seconds.
E602	First checkpoint – SSA PCI adapter open firmware has started.
E603	BIST has completed with an error.
E604	BIST and subsequent POSTs have completed successfully.
E605	BIST has completed successfully but the subsequent POSTs have failed.
E60E	SSA PCI adapter open firmware about to exit (no stack corruption).
E60F	SSA PCI adapter open firmware has run unsuccessfully. The adapter is not responding.
E6FF	SSA PCI adapter firmware about to exit (with stack corruption).

Chapter 3. PCI Interface

Characteristics

The Advanced SerialRAID Adapter complies with revision 2.1 of the PCI Local Bus specification with the exception that the execution time of BIST is 2.3 seconds. This includes:

- 32–bit address and data.
- 64K DMA paths for data transfer.
- Up to 90 MB/s sustained transfer rate as an Initiator with a burst length of 512 bytes and no wait states.
- Locking, JTAG, and cache snooping are not supported.
- BIST is supported.
- Expansion ROM is available.
- 0 — 33.33 MHz clock.
- Universal signalling. (3.3V or 5V.)
- +5V and +12V power. (+3.3V is not used.)

Target Cycles

Memory writes, I/O writes, and long reads are posted operations. The adapter may retry a target cycle if it is busy with a posted operation.

The Advanced SerialRAID Adapter will not modify a register on any PCI command that it aborts by asserting SERR# or PERR# if parity checking is enabled.

The adapter ignores null cycles (that is, when all byte enables are inactive).

When PCI RST# is asserted the adapter resets by flushing its scan chains. During this period the PCI drivers are tri-stated and the adapter will not respond to PCI accesses. The drivers are enabled about 5µs after RST# is negated. The microcode then initializes the configuration registers from flash memory. During this period the adapter will retry any PCI accesses.

Configuration Cycles

- 1, 2, 3, and 4-byte reads and writes are supported. (Any combination of byte enables.)
- Writes are not posted.
- The adapter asserts DEVSEL# only if IDSEL is active, AD(10–8) = 000b and AD(1–0) = 00b during the address phase. Otherwise the PCI command is ignored.
- The adapter issues Target Abort if the parity of AD(31–0) and C/BE#(3–0) is bad during the address phase. (SERR# is asserted if it is enabled in the PCI command register.)

Memory Cycles

- The adapter issues Target Abort if the parity of AD(31–0) and C/BE#(3–0) is bad during the address phase. (SERR# is asserted if enabled.)
- The adapter issues Target Disconnect if AD(1–0) ≠ 00b during the address phase.
- The adapter ignores the byte enables and returns 4 bytes on all reads.
- The BAR_1 window (SDRAM) supports 1, 2, 3, and 4-byte accesses. The adapter issues Target Abort for all writes. It will burst up to 128 bytes using a single 128–byte buffer for writes and a separate prefetch buffer for reads. If the requested read data is not already in the prefetch buffer the adapter will prefetch 128 bytes starting from the preceding 128-byte address boundary before transferring any data on the PCI bus.
- The BAR_2 window (registers and RAM) supports 4-byte writes to specific registers only. The adapter issues Target Abort for 1, 2, and 3-byte writes and writes to protected regions. The adapter will disconnect all bursts.
- The BAR_6 window (Expansion ROM) supports 1, 2, 3, and 4-byte reads. The adapter issues Target Abort for all writes and disconnects bursts.

I/O Cycles

- The adapter issues Target Abort if the parity of AD(31–0) and C/BE#(3–0) is bad during the address phase. (SERR# is asserted if enabled.)
- The adapter ignores the byte enables and returns 4 bytes on all reads.
- The BAR_0 window (registers and RAM) supports 4-byte writes to specific registers only. The adapter issues Target Abort for 1, 2, and 3-byte writes and writes to protected regions.

Initiator Cycles

The Advanced SerialRAID Adapter does not issue configuration or I/O cycles as an initiator.

Memory Cycles

- The Advanced SerialRAID Adapter is an initiator for all data transfers.
- The adapter will normally burst up to 512 bytes at a time, but it will not burst across a 512-byte address boundary. IRDY# is asserted for the entire burst to minimize bus occupancy.

  If possible the adapter will transfer 4 KB on behalf of a particular transaction before switching to another transaction. This helps to minimize thrashing of any caches between the PCI bus and memory.
- System cache line sizes of 64 and 128 bytes are supported.
- The adapter issues Memory Read Multiple commands if the transfer equals or exceeds a cache line.
- The adapter (if allowed) issues the Memory Write and Invalidate command if the transfer equals a cache line or a multiple of a cache line.
- By default, the adapter normally negates REQ# after it asserts IRDY#.
- The adapter does not issue back-to-back commands as an initiator.

Interrupts

The Advanced SerialRAID Adapter signals all PCI interrupts by asserting INTA#. An interrupt is generated whenever a bit is set to 1b in the Interrupt register and the corresponding bit in the Interrupt Mask register is set to 0b.

Commands

The following table shows the PCI commands that the Advanced SerialRAID Adapter issues as an Initiator and supports as a Target.

C/BE	**Command**	**Issued as Initiator**	**Supported as target**
0000b	Interrupt acknowledge	No	No
0001b	Special cycle	No	No
0010b	I/O Read	No	Yes
0011b	I/O Write	No	Yes
0100b	Reserved	No	No
0101b	Reserved	No	No
0110b	Memory read	Yes	Yes
0111b	Memory write	Yes	Yes
1000b	Reserved	No	No
1001b	Reserved	No	No
1010b	Configuration read	No	Yes
1011b	Configuration write	No	Yes
1100b	Memory read multiple	Yes	Alias to 0110b
1101b	Dual address cycle	No	No
1110b	Memory read line	Yes	Alias to 0110b
1111b	Memory write and invalidate	Yes	Alias to 0111b

PCI Configuration Registers

The following registers are accessible using Configuration Read and Configuration Write commands when IDSEL is active. They can also be read, but not written, by way of PCI memory and I/O space.

The fields of the configuration registers are mapped into PCI configuration space as follows:

Config Address	31	23	15	7 0
00h	Device ID		Vendor ID	
04h	Status		Command	
08h	Class Code			Revision ID
0Ch	BIST	Header Type	Latency Timer	Cache Line Size
10h	Base Address 0 (256 byte I/O-mapped window into registers and RAM)			
14h	Base Address 1 (8 MB memory-mapped window into SDRAM)			
18h	Base Address 2 (32 KB memory-mapped registers and RAM)			
1Ch	Base Address 3 (128 KB memory-mapped NVRAM)			
20h	Base Address 4 (1 MB memory-mapped Flash)			
24h	Base Address 5 (Reserved = 0000 0000h)			
28h	Reserved = 0000 0000h			
2Ch	Subsystem ID		Subsystem Vendor ID	
30h	Base Address 6 (16 KB Expansion ROM)			
34h	Reserved = 0000 0000h			
38h	Reserved = 0000 0000h			
3Ch	Max_Lat	Min_Gnt	Interrupt Pin	Interrupt Line
40h	Reserved			
44h	PCI Bus Control			
48h	PCI Swap Control			
4Ch	Reserved			
50–7Ch	Reserved = 0000 0000h			
80–FCh	Alias of 00–7Ch			

These registers are initialized to the values shown after a power on and a PCI reset. They are not changed by the adapter during an adapter software reset.

PCI Vendor ID Register

The PCI vendor ID register identifies the vendor of the adapter card.

PCI configuration address	00h
PCI memory address	BAR_2 + 00h (Read only)
PCI I/O address	BAR_0 + 00h + 00h (Read only)
Reset value	1014h

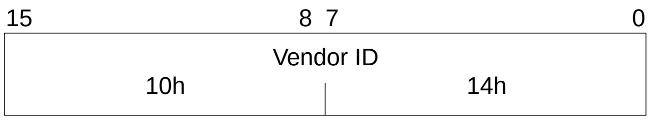


Figure 6. PCI Vendor ID Register

The register cannot be written from the PCI bus.

Table 21. PCI Vendor ID Register

Bits	Name	Description	Type
15–0	VendorID	**Vendor ID.** Identifies the manufacturer of the adapter. These bits are set to 1014h to identify the manufacturer as IBM.	R

PCI Device ID Register

The PCI device ID register uniquely identifies the particular PCI adapter.

PCI configuration address	02h
PCI memory address	BAR_2 + 02h (Read only)
PCI I/O address	BAR_0 + 00h + 02h (Read only)
Reset value	0058h

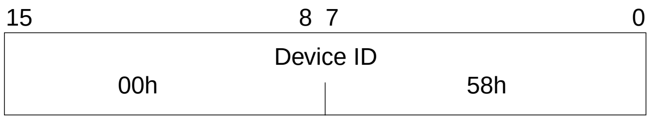


Figure 7. PCI Device ID Register

Table 22. PCI Configuration ID Register

Bits	Name	Description	Type
15–0	DeviceID	**Device ID.** Identifies the PCI device. These bits are set to 0058h to identify the adapter as a Advanced SerialRAID Adapter	R

PCI Command Register

The configuration software writes this register to control the adapter's ability to generate and respond to PCI cycles. When the register is 0000h, the adapter is logically disconnected from the PCI bus for all functions except Configuration cycles as a Target.

PCI configuration address	04h
PCI memory address	BAR_2 + 04h (Read only)
PCI I/O address	BAR_0 + 00h + 04h (Read only)
Reset value	0000h

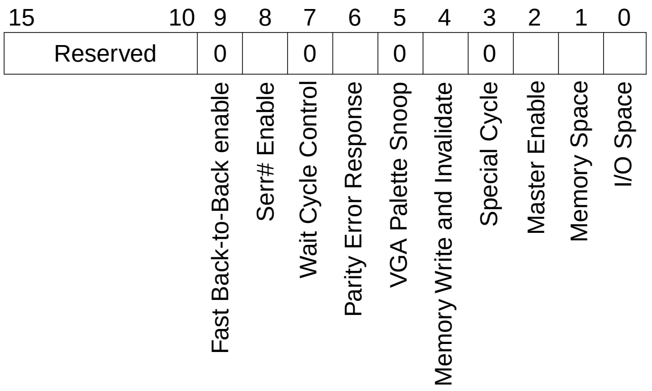


Figure 8. PCI Command Register

Table 23. PCI Command Register

Bits	Name	Description	Type
15–10	-	Reserved	R
9	-	**Fast Back-to-Back Enable.** This bit will always read as 0b.	R
8	SERRenb	**SERR# Enable.** The configuration software may set this bit to 1b to enable the SERR# driver. When set to 0b, the SERR# driver is disabled. Both this bit and PERRenb must be set to 1b to report address parity errors.	R/W
7	-	**Wait Cycle Control.** This bit is always read as 0b.	R
6	PERRenb	**Parity Error Response.** The configuration software may set this bit to 1b to enable checking of address and data parity checking. When set to 0b, parity errors are ignored.	R/W
5	-	**VGA Palette Snoop.** This bit is always read as 0b.	R
4	MemWrInv	**Memory Write and Invalidate.** The configuration software may set this bit to 1b to enable the adapter to issue Memory Write and Invalidate commands as a Master.	R/W

Table 23. PCI Command Register (continued)

Bits	Name	Description	Type
3	-	**Special Cycle.** This bit is always read as 0b.	R
2	Master	**Master.** The configuration software must set this bit to 1b to allow the adapter to behave as a bus master.	R/W
1	MemSpace	**Memory Space.** The configuration software must set this bit to 1b to allow the adapter to respond to memory space accesses as a Target.	R/W
0	IOspace	**I/O Space.** The configuration software must set this bit to 1b to allow the adapter to respond to I/O space accesses as a Target.	R/W

PCI Status Register

The PCI status register is used to report certain events on the PCI bus.

PCI configuration address	06h
PCI memory address	BAR_2 + 06h (Read only)
PCI I/O address	BAR_0 + 00h + 06h (Read only)
Reset value	0200h

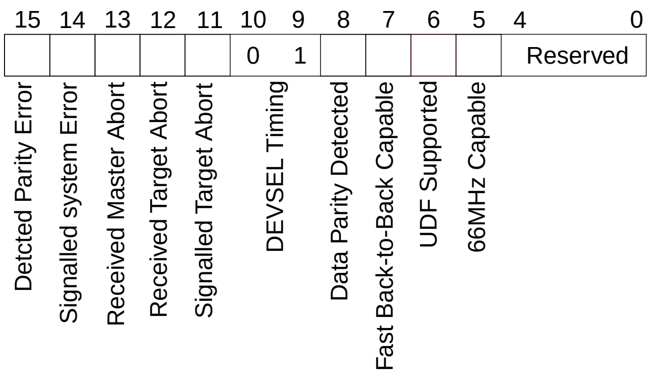


Figure 9. PCI Status Register

Table 24. PCI Status Register

Bits	Name	Description	Type
15	DetPERR	**Detected Parity Error.** When this bit is set to 1b, the adapter has detected a parity error as an Initiator or a Target, either on Read Data or with a PERR#. This bit is set regardless of the value of PERRenb in the PCI Command register.	R/C

Table 24. PCI Status Register (continued)

Bits	Name	Description	Type
14	SigSERR	**Signaled System Error.** When this bit is set to 1b, the adapter has asserted SERR# to report an address parity error. This requires that PERRenb is set to 1b and SERRenb is set to 1b in the PCI Command register.	R/C
13	RxMabort	**Received Master Abort.** The adapter sets this bit to 1b as a Master if it terminates a PCI cycle with Master Abort.	R/C
12	RxTabort	**Received Target Abort.** The adapter sets this bit to 1b if it receives Target Abort as a Master.	R/C
11	SigTabort	**Signaled Target Abort.** The adapter sets this bit to 1b if it signals Target abort as a Target.	R/C
10–9	DevSel	**DEVSEL Timing.** These bits indicate the slowest time that the adapter asserts DEVSEL# for any command except Configuration Read and Configuration Write. (DevSel = 01b indicates medium timing.)	R
8	DataPar	**Data Parity Detected.** The adapter sets this bit to 1b when PERRenb = 1b in the Command register and it detects a parity error as a Master either on read data or through PERR#.	R/C
7	-	**Fast Back-to-Back Capable.** This bit will always read as 0b.	R
6	-	**UDF Supported.** This bit will always read as 0b.	R
5	-	**66 MHz capable.** This bit will always read as 0b.	R
4–0	-	Reserved	R

PCI Revision ID Register

The PCI Revision ID register identifies the revision level of the adapter.

PCI configuration address	08h
PCI memory address	BAR_2 + 08h (Read only)
PCI I/O address	BAR_0 + 00h + 08h (Read only)
Reset value	040h

The register cannot be written from the PCI bus.

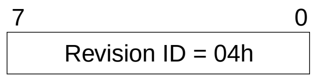


Figure 10. PCI Revision ID Register

Table 25. PCI Revision ID register

Bits	Name	Description	Type
7–0	RevID	**Revision ID.** This is an unsigned binary integer that describes the level of the microcode.	R

PCI Class Code Register

The PCI Class Code register identifies the generic function of the adapter.

PCI configuration address	09h
PCI memory address	BAR_2 + 09h (Read only)
PCI I/O address	BAR_0 + 09 + 00h (Read only)
Reset value	00 02 0Ch (Expressed as a character string)

23 16	15 8	7 0
Base Class 0Ch	Sub Class 02h	Prog I/F 00h

Figure 11. PCI Class Code Register

Table 26. PCI Class Code Register

Bits	**Name**	**Description**	**Type**
23–16	BaseClass	**Base Class Encoding.** This field is set to 0Ch to identify that the adapter is a serial-bus controller.	R
15–8	SubClass	**Subclass Encoding.** This field is set to 02h to indicate that the adapter is an SSA adapter subclass.	R
7–0	ProgIF	**Programming Interface.** This field is 00h because the adapter does not conform to a standard programming interface.	R

PCI Cache Line Size Register

The PCI Cache Line Size register specifies the system cache line size.

PCI configuration address	0Ch
PCI memory address	BAR_2 + 0Ch (Read only)
PCI I/O address	BAR_0 + 00h + 0Ch (Read only)
Reset value	00h

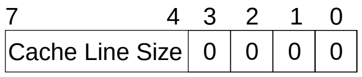


Figure 12. PCI Cache Line Size Register

Table 27. PCI Cache Line SizeRegister

Bits	Name	Description	Type
7–4	LineSize	**Cache Line size.** The configuration software writes this field with the number of bytes in a system cache line divided by 16. Only line sizes of 128 or 64 bytes are supported. If any other value is loaded, the adapter will only issue Memory Read and Memory Write commands as a Master.	R/W
3–0	-	These bits are always set to 0h	R

PCI Latency Timer Register

The PCI Latency Timer register specifies, in units of PCI bus clocks, how long the adapter can burst data on the PCI bus as a Master.

PCI configuration address	0Dh
PCI memory address	BAR_2 + 0Dh (Read only)
PCI I/O address	BAR_0 + 00h + 0Dh (Read only)
Reset value	00h

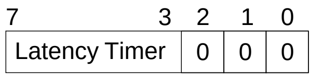


Figure 13. PCI Latency Timer Register

Table 28. PCI Latency Timer Register

Bits	Name	Description	Type
7–3	Latency	**Latency timer.** The adapter will burst data on the PCI bus for 8 times the number of clocks specified in this field before allowing pre-emption. If the arbiter has negated GNT# and the current burst has exceed the latency time, then the adapter will get off the bus as soon as possible. If the Latency timer is set to zero and GNT# is negated the adapter will terminate the current burst at the next 128-byte address boundary.	R/W
2–0	-	These bits are always read as 000b.	R

PCI Header Type Register

The PCI Header register identifies the layout of bytes 10 to 3Fh in the configuration space.

PCI configuration address	0Eh
PCI memory address	BAR_2 + 0Eh (Read only)
PCI I/O address	BAR_0 + 00h + 0Eh (Read only)
Reset value	00h

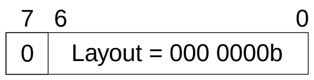


Figure 14. PCI Header Type Register

Table 29. PCI Header Type Register

Bits	Name	Description	Type
7	-	**Multiple functions.** This bit will always be 0b to indicate that the adapter does not have multiple functions,	R
6–0	-	**Layout.** This field will always read as 000 0000b to indicate that the configuration header has the standard layout.	R

PCI Built-In Self Test Register

The PCI Built-In Self Test register controls the execution of built-in self tests (BIST). It also reports POST failures.

PCI configuration address	0Fh
PCI memory address	BAR_2 + 0Fh (Read only)
PCI I/O address	BAR_0 + 00h + 0Fh (Read only)
Reset value	80h

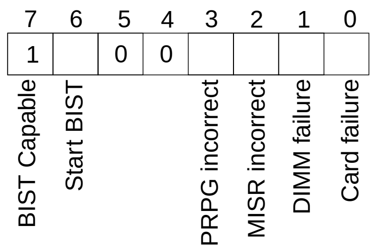


Figure 15. PCI Built-In Self Test Register

Table 30. PCI Built-In Self Test (BIST) Register

Bits	Name	Description	Type
7	BISTcap	**BIST Capable.** This bit is set to 1b to indicate that the adapter is capable of BIST.	R
6	StartBIST	**Start BIST.** This bit is written with 1b from the PCI bus to invoke BIST. The host must then wait for 2.3 seconds for BIST to complete before accessing the adapter again. The adapter resets StartBIST to 0b when BIST is complete. The adapter does not perform a full BIST the first time StartBIST is set following PCI RST#.	R/S
5–4	-	Reserved. These bits are always set to 00b.	R
3	PRPGerr	**PRPG Error.** The adapter sets this bit to 1b on completion of BIST if the PRPG value is incorrect	R
2	MISRerr	**MISR Error.** The adapter sets this bit to 1b on completion of BIST if the MISR value is incorrect	R
1	DIMMfail	**DIMM failure.** The microcode sets this bit to 1b if it detects a failure of the SDRAM DIMM during POST.	R
0	Cardfail	**Card failure.** The microcode sets this bit to 1b if it detects a failure of the base card during POST.	R

PCI Base Address Register 0 (BAR_0)

This register provides a 256–byte window in PCI I/O space for accessing the adapter registers and adapter RAM. The position of the window can be moved within the adapter 32 KB register or RAM space by programming BAR_0 offset in the PCI Bus Control register.

PCI configuration address	10h
PCI memory address	BAR_2 + 10h (Read only)
PCI I/O address	BAR_0 + 00h + 10h (Read only)
Reset value	0000 0001h

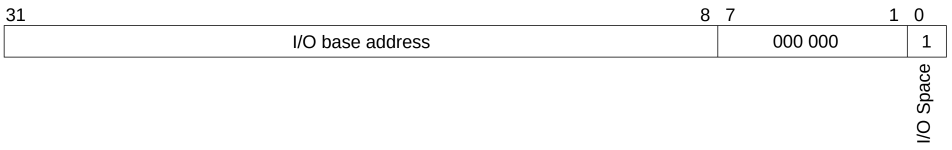


Figure 16. Base Address Register 0 (IO-mapped registers and RAM)

Table 31. Base Address Register 0 (IO-mapped registers and RAM)

Bits	Name	Description	Type
31–8	Base	**Base address.** These bits are programmed by the configuration software with the base address of the window in PCI I/O space.	R/W
7–1	-	**Reserved.** These bits are always set to 0000 000b.	R
0	IOspace	**I/O Space Indicator.** This bit always reads as 1b to indicate that the window should be mapped into PCI I/O space.	R

The BAR_0 I/O window is read-only except for the following words:

Table 32. Read/Write registers in BAR_0 and BAR_2 windows:

Offset	Register
10Ch	BIST Control
114h	RRIN
13Ch	Adapter error
1A4h	(Reserved)
1C0h	Doorbell
1E0h	Interrupt (Read/Set)
1E4h	Interrupt (Clear)
1E8h	Interrupt Mask (Read/Set)
1ECh	Interrupt Mask (Clear)

The adapter will issue Target Abort for a write access to any other address.

PCI Base Address Register 1 (BAR_1)

This register maps an 8MB SDRAM window into PCI memory space. The adapter will issue Target Abort for all writes.

PCI configuration address	14h
PCI memory address	BAR_2 + 14h (Read only)
PCI I/O address	BAR_0 + 14h (Read only)
Reset value	0000 0008h

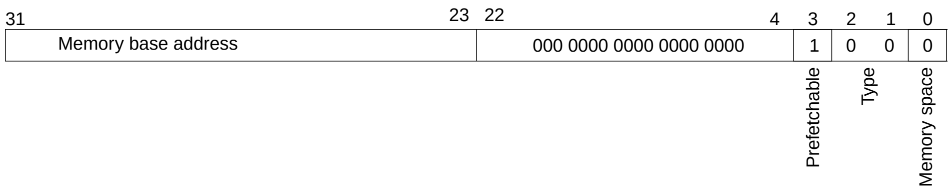


Figure 17. Base Address Register 1 (SDRAM window)

Table 33. Base Address Register 1 (SDRAM window)

Bits	Name	Description	Type
31–23	Base	**Base address.** These bits are programmed by the configuration software with the base address of the window in PCI I/O space.	R/W
22–4	-	**Reserved.** These bits are always set to 000 0000 0000 0000 0000b.	R
3	Prefetch	**Prefetchable.** This bit always reads as 1b to indicate that there are no side effects on Reads. Note that the setting of this bit does *not* indicate that the BAR_1 memory space is cacheable. (The adapter may change the contents of SDRAM.)	R
2–1	Type	**Type.** These bits are always set to 00b to indicate that the base register is 32 bits wide and the window can be mapped anywhere in the 32-bit PCI address space.	R
0	Memory	**Memory Space Indicator.** This bit is always set to 0b to indicate that the window should be mapped into PCI memory space.	R

The BAR_1 window provides access to the Diagnostic Area which contains the adapter VPD and diagnostic information.

PCI Base Address Register 2 (BAR_2)

This register maps the adapter registers and on-chip RAM into a 32 KB region in PCI memory space. Except for the registers listed in Table 32 on page 58, the adapter will issue Target Abort for all writes to the BAR_2 memory window.

PCI configuration address	18h
PCI memory address	BAR_2 + 18h (Read only)
PCI I/O address	BAR_0 + 00h + 18h (Read only)
Reset value	0000 0008h

Memory-mapped registers and RAM

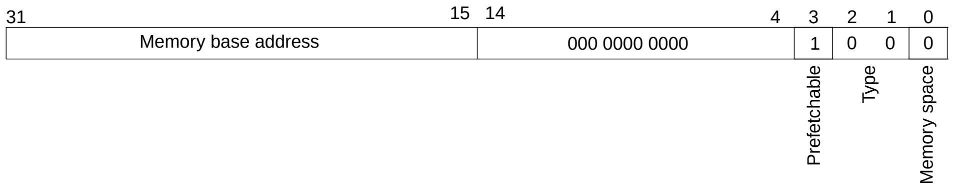


Figure 18. Base Address Register 2 (Memory-mapped registers and RAM)

Table 34. Base Address Register 2(Memory-mapped registers and RAM)

Bits	Name	Description	Type
31–15	Base	**Base address.** These bits are programmed by the configuration software with the base address of the window in PCI memory space.	R/W
14–4	-	**Reserved.** These bits are always set to 000 0000 0000b.	R
3	Prefetch	**Prefetchable.** This bit always reads as 1b to indicate that there are no side effects on Reads. Note that the setting of this bit does *not* indicate that the BAR_2 memory space is cacheable. (The adapter may change the contents of registers and RAM.)	R
2–1	Type	**Type.** These bits are always set to 00b to indicate that the base register is 32 bits wide and the registers can be mapped anywhere in the 32-bit PCI address space.	R
0	Memory	**Memory Space Indicator.** This bit is always set to 0b to indicate that the registers should be mapped into PCI memory space.	R

PCI Base Address Register 3 (BAR_3)

This register maps the NVRAM into a 128 KB region in PCI memory space. The adapter issues Target Abort for all writes to the window.

PCI configuration address	1Ch
PCI memory address	BAR_2 + 1Ch (Read only)
PCI I/O address	BAR_0 + 00h + 1Ch (Read only)
Reset value	0000 0008h

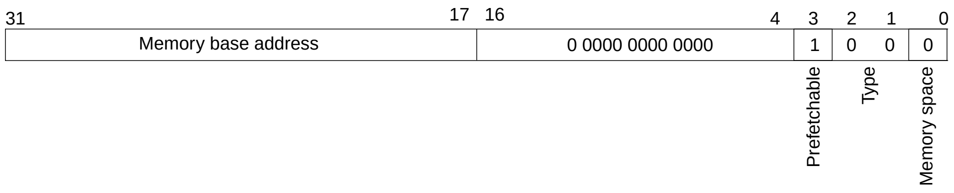


Figure 19. Base Address Register 3 (NVRAM)

Table 35. Base Address Register 3 (NVRAM)

Bits	Name	Description	Type
31–17	Base	**Base address.** These bits are programmed by the configuration software with the base address of the NVRAM in PCI memory space.	R/W
16–4	-	**Reserved.** These bits are always set to 0 0000 0000 0000b.	R
3	Prefetch	**Prefetchable.** This bit always reads as 1b to indicate that there are no side effects on Reads. Note that the setting of this bit does *not* indicate that the BAR_3 memory space is cacheable. (The adapter may change the NVRAM.)	R
2–1	Type	**Type.** These bits are always set to 00b to indicate that the base register is 32 bits wide and the NVRAM can be mapped anywhere in the 32-bit PCI address space.	R
0	Memory	**Memory Space Indicator.** This bit is always set to 0b to indicate that the NVRAM should be mapped into PCI memory space.	R

The adapter does not use the BAR_3 window for communication with the host system.

PCI Base Address Register 4 (BAR_4)

This register maps the first 1 MB of flash memory into PCI memory space. The adapter issues Target Abort for all writes to the window.

PCI configuration address	20h
PCI memory address	BAR_2 + 20h (Read only)
PCI I/O address	BAR_0 + 00h + 20h (Read only)
Reset value	0000 0008h

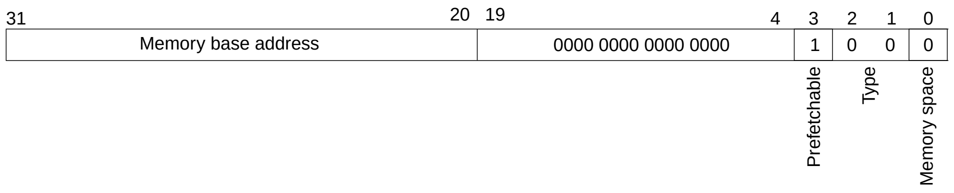


Figure 20. Base Address Register 4 (Flash memory)

Table 36. Base Address Register 4(Flash memory)

Bits	Name	Description	Type
31–21	Base	**Base address.** These bits are programmed by the configuration software with the base address of the flash memory in PCI memory space.	R/W
20–4	-	**Reserved.** These bits are always set to 0 0000 0000 0000 0000b.	R
3	Prefetch	**Prefetchable.** This bit always reads as 1b to indicate that there are no side effects on Reads. Note that the setting of this bit does *not* indicate that the BAR_4 memory space is cacheable. (The adapter may change the contents of the Flash memory.)	R
2–1	Type	**Type.** These bits are always set to 00b to indicate that the base register is 32 bits wide and the flash memory can be mapped anywhere in the 32-bit PCI address space.	R
0	Memory	**Memory Space Indicator.** This bit is always set to 0b to indicate that the flash memory should be mapped into PCI memory space.	R

The adapter does not use the BAR_4 window for communication with the host system.

PCI Subsystem Vendor ID Register

PCI configuration address	2Ch
PCI memory address	BAR_2 + 2Ch (Read only)

PCI I/O address	BAR_0 + 00h + 2Ch (Read only)
Reset value	1014h

Subsystem Vendor ID Register

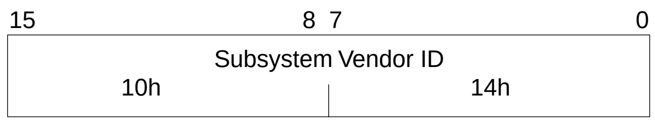


Table 37. Subsystem vendor ID Register

Bits	**Name**	**Description**	**Type**
15–0	SSVendorID	**Subsystem Vendor ID.** This bit is set to 1014h to identify the manufacturer of the adapter card as IBM.	R

PCI Subsystem ID Register

The configuration software reads this register to identify the particular adapter card.

PCI configuration address	2Eh
PCI memory address	BAR_2 + 2Eh (Read only)
PCI I/O address	BAR_0 + 00h + 2Eh (Read only)
Reset value	0091h

Subsystem ID Register

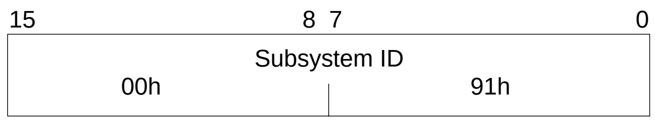


Table 38. Subsystem ID Register

Bits	**Name**	**Description**	**Type**
15–0	SSID	**Subsystem ID.** This bit is set to 0091h to identify the adapter card as an Advanced SerialRAID Adapter.	R

PCI Base Address Register 6 (BAR_6)

This register maps the 32 KB expansion ROM into PCI memory space. The adapter will issue Target Abort for all writes to the window.

PCI configuration address	30h
PCI memory address	BAR_2 + 30h (Read only)
PCI I/O address	BAR_0 + 00h + 30h (Read only)
Reset value	0000 0000h

Base Address Register 6

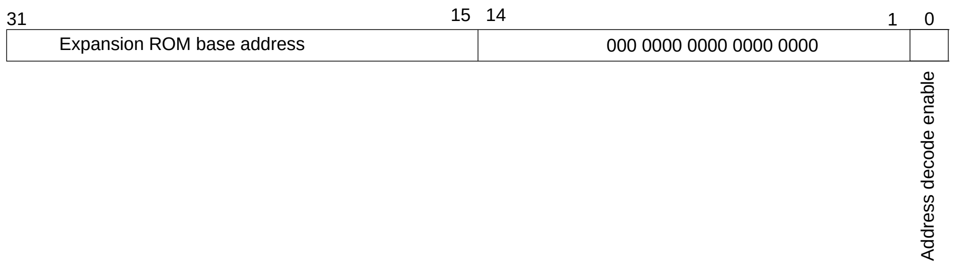


Table 39. Base Address Register 6 (Expansion ROM)

Bits	Name	Description	Type
31–15	Base	**Base Address.** These bits are programmed by the configuration software with the base address of the expansion ROM in PCI memory space.	R/W
14–1	-	Reserved	R
0	ADenb	**Address Decode Enable.** The configuration software may set this bit to 1b to enable accesses to the expansion ROM. A value of 0b disables the expansion ROM space.	R/W

The BAR_6 window provides access to the 2 images containing the CBIOS and the Open Firmware.

PCI Interrupt Line Register

This register indicates the input of the system interrupt controller that is connected to the adapter’s PCI interrupt pin. It is written by the configuration software and read by the operating system and device driver. The contents are not used by the microcode.

PCI configuration address	3Ch
PCI memory address	BAR_2 + 3Ch (Read only)
PCI I/O address	BAR_0 + 00h + 3Ch (Read only)
Reset value	00h

PCI Interrupt Line Register

7   0
Interrupt Line

Table 40. PCI Interrupt Line Register

Bits	Name	Description	Type
7–0	IntLine	**Interrupt Line.** These bits are written by the configuration software to indicate which input of the system interrupt controller is connected to the adapter. These values are system specific.	R/W

PCI Interrupt Pin Register

This register is read by the configuration software to determine which PCI interrupt pin is used by the adapter.

PCI configuration address	3Dh
PCI memory address	BAR_2 + 3Dh (Read only)
PCI I/O address	BAR_0 + 00h + 3Dh (Read only)
Reset value	01h

PCI Interrupt Pin Register

7   0
Interrupt Pin = 01h

Table 41. PCI Interrupt Pin Register

Bits	Name	Description	Type
7–0	IntPin	**Interrupt Pin.** These bits are always set to 01h to indicate that the adapter uses the INTA# pin.	R

PCI Min_Gnt Register

This register specifies the minimum PCI burst length required by the adapter.

PCI configuration address	3Eh
PCI memory address	BAR_2 + 3Eh (Read only)
PCI I/O address	BAR_0 + 00h + 3Eh (Read only)
Reset value	00h

PCI Min_Gnt Register

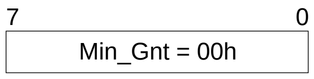


Table 42. PCI Min_Gnt Register

Bits	Name	Description	Type
7–0	MinGnt	**Minimum Grant.** These bits are always set to 00h to indicate that the adapter has no specific requirement for burst length.	R

PCI Max_Lat Register

This register specifies how often the adapter need to gain access to the PCI bus.

PCI configuration address	3Fh
PCI memory address	BAR_2 + 3Fh (Read only)
PCI I/O address	BAR_0 + 00h + 3Fh (Read only)
Reset value	00h

PCI Max_Lat Register

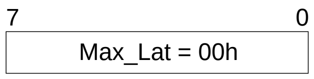


Table 43. PCI Max_Lat Register

Bits	Name	Description	Type
7–0	MaxLat	**Maximum latency.** These bits are always set to 00h to indicate that the adapter has no specific requirement for access latency.	R

PCI Bus Control Register

This register contains miscellaneous controls for the adapter PCI interface.

PCI configuration address	44h
PCI memory address	BAR_2 + 44h (Read only)
PCI I/O address	BAR_0 + 00h + 44h (Read only)
Reset value	0000 0100h

PCI Bus control Register

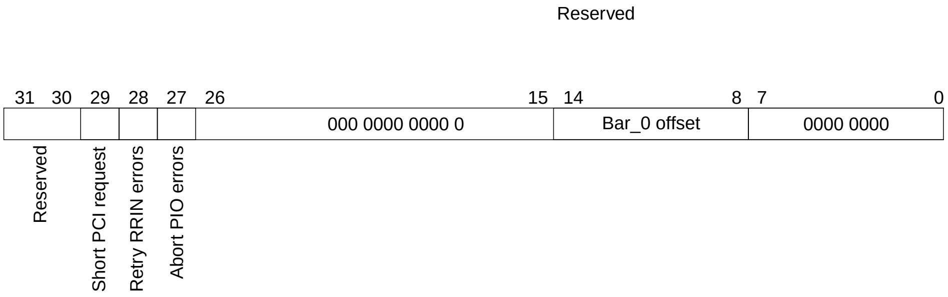


Table 44. PCI Bus Control Register

Bits	Name	Description	Type
31–30	-	**Reserved** These bits must be set to 00b.	R/W
29	ShortReq	**Short PCI request:** • When ShortReq = 0b (default) the adapter will assert REQ# during DMA until it negates FRAME#. • When ShortReq = 1b, the adapter will assert REQ# during DMA only until it asserts IRDY#. This behavior is required by some PC applications. In this mode the burst size is controlled by the PCI Latency Timer register since GNT# will always be negated early. If the latency timer is set too small the DMA bandwidth will be degraded by frequent pre-emptive disconnects.	R/W
28	RetryRRIN	**Retry RRIN errors:** • When set to 1b, the adapter will retry a PCI write to the RRIN register if the RRIN queue is full or disabled. • When set to 0b, the adapter will discard a PCI write to the RRIN register if the RRIN queue is full or disabled.	R/W
27	AbortPIO	**Abort PIO read errors:** If the adapter detects a PIO error it sets the Error bit in the PCI interrupt register. This bit controls how a read cycle is terminated: • When AbortPIO = 1b, the adapter will issue TRDY# to end the PIO normally. • When AbortPIO = 0b, the adapter will issue Target Abort.	R/W
26–15	-	Reserved. These bits are always set to 000 0000 0000 0b	R

Table 44. PCI Bus Control Register (continued)

Bits	Name	Description	Type
14–8	BAR0offset	**BAR_0 offset:** These bits are concatenated with address bits 7–2 from the PCI bus for I/O accesses with BAR_0. This allows access to the full 32 KB address range of the adapter registers and RAM, even though BAR_0 can only access 256 bytes at a time.	R/W
7–0	-	Reserved. These bits are always set to 0000 0000b	R

PCI Swap Control Register

This register allows a byte swap function in the adapter local processor to be selectively enabled for each Base Address register. When enabled, the byte order is reversed within a word for Target accesses, for example for a Big-Endian host.

PCI configuration address	48h
PCI memory address	BAR_2 + 48h (Read only)
PCI I/O address	BAR_0 + 00h + 48h (Read only)
Reset value	0000 0000h

PCI Swap Control Register

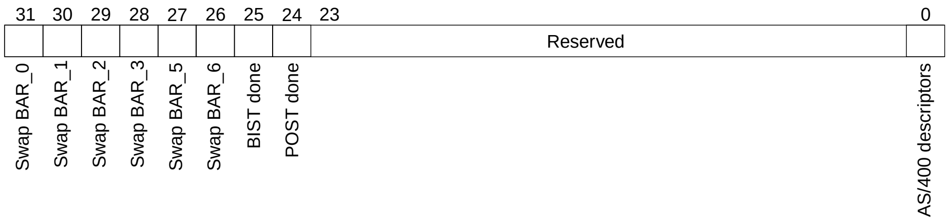


Table 45. PCI Swap Control Register

Bits	Name	Description	Type
31	SwapBAR0	**Swap BAR_0.** When set to 1b, the adapter swaps the byte order for Target accesses using BAR_0.	R/W
30	SwapBAR1	**Swap BAR_1.** When set to 1b, the adapter swaps the byte order for Target accesses using BAR_1.	R/W
29	SwapBAR2	**Swap BAR_2.** When set to 1b, the adapter swaps the byte order for Target accesses using BAR_2.	R/W
28	SwapBAR3	**Swap BAR_3.** When set to 1b, the adapter swaps the byte order for Target accesses using BAR_3.	R/W
27	SwapBAR4	**Swap BAR_4.** When set to 1b, the adapter swaps the byte order for Target accesses using BAR_4.	R/W

Table 45. PCI Swap Control Register (continued)

Bits	Name	Description	Type
26	SwapBAR6	**Swap BAR_6.** When set to 1b, the adapter swaps the byte order for Target accesses using BAR_6.	R/W
25	BISTdone	**BIST done.** Microcode sets this bit to 1b when the adapter BIST has been run.	R/W
24	POSTdone	**POST done.** Microcode sets this bit to 1b when it has completed the POST. This indicates that the VPD has been stored in SDRAM and that it is safe for the host software to read it.	R/W
23–1	-	Reserved	R/W
0	Reserved	The host software must not set this bit to 1b.	R/W

Communication Registers

These registers are mapped into PCI memory space through BAR_2. They can also be accessed 256 bytes at a time in PCI I/O space through BAR_0.

M/IO Offset	Register
10Ch	BIST Control
110h	Configuration/Status
114h	RRIN
13Ch	Adapter Error
1C0h	Doorbell
1E0h	Interrupt (Read/Set)
1E4h	Interrupt (Clear)
1E8h	Interrupt Mask (Read/Set)
1ECh	Interrupt Mask (Clear)
1F0h	PIO Error Address (Clear)

BIST Control Register

This register contains status bits for the adapter Built-In Self-Test. It also has various reset functions.

PCI memory address	BAR_2 + 10h (Read/Set or Clear)
PCI I/O address	BAR_0 + 100h + 0Ch (Read/Set or Clear)
PCI reset value	F700 0000h
BIST complete value	C700 0000h
Software reset value	9700 0000h

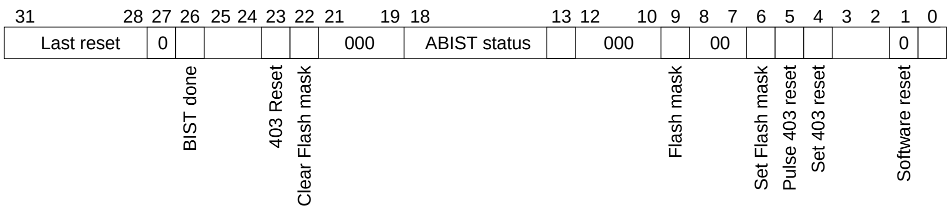


Figure 21. BIST Control Register

Table 46. BIST Control register

Bits	**Name**	**Description**	**Type**
31–28	LastRst	**Last reset.** The adapter sets these 4 bits to show the cause of the last reset: • 1001b - Software reset • 1100b - PCI Configuration started BIST • 1111b - PCI RST#	R
27	-	Reserved. This bit is always set to 0b.	R
26	BISTdone	**BIST done.** The adapter sets this bit to 1b when BIST or a Software reset completes. The host software may reset BISTdone by writing 1b to this bit.	R/C
25–24	-	Reserved. The host software should not write 1b to either of these bits.	R/C
23	Rst403	**403 reset**The host software may release the 403GCX reset by writing 1b to this bit. Rst403 reads as 1b when the 403GCX reset signal is active.	R/C
22	ClrFlashMsk	**Clear Flash mask.** The host software may clear FlashMsk to 0b by writing 1b to this bit. ClrFlashMsk always reads as 0b.	C
21–19	-	Reserved = 000b	R
18	CacheDone	**Cache Done.** The adapter sets this bit to 1b when the ABIST for the L2 cache RAM completes successfully.	R
17	CCtlDone	**Cache Control Done.** The adapter sets this bit to 1b when the ABIST for the L2 cache control RAM completes successfully.	R
16	CmdCDone	**Command Cache Done.** The adapter sets this bit to 1b when the ABIST for the command cache RAM completes successfully.	R
15	CacheErr	**Cache Error.** The adapter sets this bit to 1b when the ABIST for the L2 cache RAM fails.	R
14	CCtlErr	**Cache Control Error.** The adapter sets this bit to 1b when the ABIST for the L2 cache control RAM fails.	R
13	CmdCErr	**Command Cache Error.** The adapter sets this bit to 1b when the ABIST for the command cache RAM fails.	R
12–10	-	Reserved = 000b.	R

Table 46. BIST Control register (continued)

Bits	Name	Description	Type
9	FlashMsk	**Flash Mask.** This bit indicates whether the Flash reset signal is masked. If FlashMsk = 0b then the Flash memory is reset by the 403 reset signal. If FlashMsk = 1b then the Flash reset signal is disabled. The host software uses this function to reprogram the Flash memory: 1. Assert the 403GCX reset by writing 1b to Set403Rst. 2. Set FlashMsk to 1b by writing 1b to SetFlashMsk. 3. Write new data to Flash memory through BAR_4. 4. Clear FlashMsk to 0b by writing 1b to bit ClrFlashMsk. 5. Release the 403GCX reset by writing 1b to bit Rst403.	R
8–7	-	Reserved = 00b	R
6	SetFlashMsk	**Set Flash Mask.** The host software may set FlashMsk to 1b by writing 1b to this bit. SetFlashMsk always reads as 0b.	S
5	PulseRst	**Pulse 403 Reset.** The host software may assert the 403GCX reset for 3 microseconds by writing 1b to this bit. PulseRst always reads as 0b.	S
4	Set403Rst	**Set 403 Reset.** The host software may set Rst403 to 1b by writing 1b to this bit. In turn this asserts the reset input to the 403GCX. Set403Rst always reads as 0b.	S
3–2	-	Reserved. the host software should not write 1b to any of these bits.	S
1	SoftRst	**Software Reset.** The host software should set this bit to 1b to reset the adapter, for instance, after a catastrophic error. The entire adapter is reset including the local processor, the local PCI buses and the SSA ports but excluding the PCI Configuration registers and data cached in the Fast write cache. The adapter BIST does not run. However, BISTdone is set to 1b when the reset is complete. SoftRst always reads as 0b.	S
0	StartBIST	Setting this bit to 1b will start the adapter BIST. The adapter will set BISTdone to 1b when BIST completes. StartBIST always reads as 0b.	S

Configuration/Status Register

This register indicates when the adapter microcode is ready to receive the first command after a reset.

PCI memory address	BAR_2 + 110h (Read only)
PCI I/O address	BAR_0 + 100h + 10h (Read only)
Reset complete value	uuuu uuuu uu1u uuuu uuuu uuuu uuuu uuuub
	(where u is undefined)

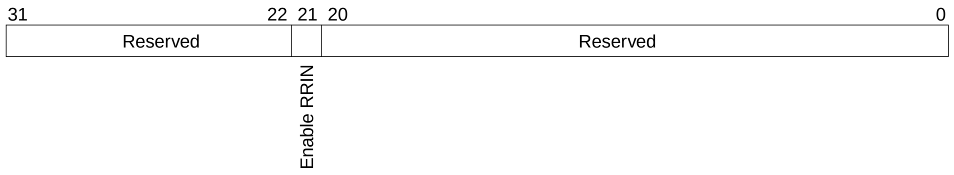


Figure 22. Configuration/Status Register

Table 47. Configuration/Status Register

Bits	**name**	**Description**	**Type**
31–22	-	Reserved	
21	EnRRIN	**Enable RRIN register.** The adapter hardware sets this bit to 0b when the adapter is reset by PCI RST#, Software Reset, or Start BIST. The adapter microcode sets EnRRIN to 1b to enable access to the RRIN register after the reset is complete. This event may be used to detect when the adapter is ready to receive Commands and transactions.	R
20–0	-	Reserved.	

RRIN Register

This is a 4–byte register that the device driver writes to issue a Request or Reply to the adapter.

PCI memory address	BAR_2 + 114h (Read/Write)
PCI I/O address	BAR_0 + 100h + 14h (Read/Write)
Reset value	0000 0000h

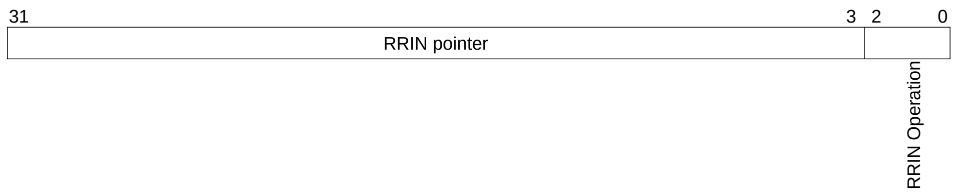


Figure 23. RRIN Register

Table 48. RRIN Register

Bits	Name	Description	Type
31–3	RRINptr	**RRIN Pointer.** This is a pointer to a GTCB or the Parameter block for a Command, or a reply_handle, as defined in “RRIN register” on page 19.	R/W
2–0	RRINop	**RRIN Operation.** This 3-bit field defines the operation to be performed by the adapter, as defined in “RRIN register” on page 19.	R/W

When the device driver attempts to write to the RRIN register the following operations occur:

1. If the RRIN queue is full or disabled then:
   - If RetryRRIN = 1b in the PCI Bus Control register then the entry will remain in the RRIN register. The adapter retries subsequent PCI writes to this register until the RRIN queue is not full and enabled.
   - If RetryRRIN = 0b, then the entry is discarded and not processed further. The adapter also sets the RRINlost bit to 1b in the PCI Interrupt register and disables the RRIN queue. Subsequent PCI writes to the RRIN register will then be discarded until the device driver resets the adapter.
2. Otherwise the adapter enters the value of the RRIN register into the RRIN queue for processing.

Adapter Error Register

This register identifies an error detected by the adapter such as host programming errors, adapter microcode errors and adapter hardware errors. It is also used to return the CBIOS Status byte.

PCI memory address	BAR_2 + 13Ch (Read/Write)
PCI I/O address	BAR_0 + 100h + 3Ch (Read/Write)
Reset value	Undefined

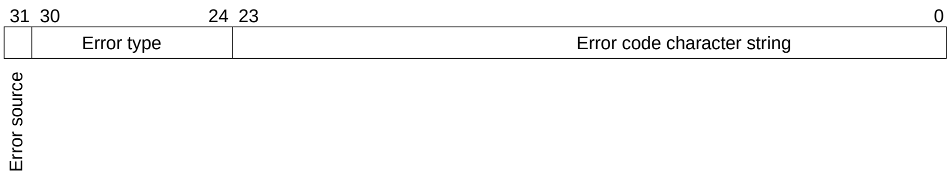


Figure 24. Adapter Error Register

The format of the error code is that, for example if the error code is 203040h, 20h is stored at address 13Ch.

Table 49. Adapter Error Register

Bits	Name	Description	Type
31	Source	**Error Source.** This bit indicates the source of a unrecoverable error code: • When Source = 1b the error originated from within the IPN kernel. • When Source = 0b the error originated outside of the IPN kernel.	R/W
30–24	ErrType	**Error Type.** This field indicates the particular error, as defined below.	R/W
23–0	ErrCode	**Adapter Error Code.** This field is used to communicate an error code to the host software when POST fails and the adapter microcode cannot run correctly. The possible codes are defined in “Adapter Error Logging Data” on page 310.	R/W

Table 50. Adapter Error Types

Code	Error Type	Description
01h	SS_INSANE	Adapter error: Insane trap
02h	SS_WRONG_INT	An ’impossible’ interrupt occurred or the host issued a second Initialize command.
03h	SS_WRONG_INI_PARMS	Host error in Initialize Command parameters
04h	SS_NOT_INIT_CMD	Host error: Command was not Initialize
05h	SS_PARMS_NOT_INLINE	Host error: All parameters should be Inline
06h	SS_TOO_MANY_REQS	Host error: Too many simultaneous requests
07h	SS_MIAMI_DMA_FAILED	Host error: Host disabled DMA in progress
08h	SS_NOT_IMPLEMENTED	Microcode not implemented
09h	SS_KNL_TRAP	Kernel detected error
0Ah	SS_KNL_INSANE	Kernel detected ’insane’ error
0Bh	SS_PARMDDRTYPE_INVALID	Parameter DDR should be DT_PCI or DT_Null
0Ch	SS_NOT_DNLD_CMD	Not Download command (during Download reset)
0Dh	SS_DNLD_TOO_BIG	Download image too big
0Eh	SS_DNLD_TOO_MANY_SG_ELS	Download has too many scatter/gather elements

Table 50. Adapter Error Types (continued)

Code	Error Type	Description
0Fh	SS_DNLD_SGLN_MISMATCH	Download scatter/gather elements don't add up
10h	SS_DNLD_LRC_FAILURE	Checksum failure in Download image
11h	SS_SIC_CLASS1	A drive indicated a class 1 error
12h	SS_WRONG_XIO_OPCODE	Invalid operation requested in Execute I/O
13h	SS_ASSERT	An assert has been hit
14h	SS_DBG_STOP	Show-stop due to debug service
17h	SS_SIC_DMA_FAILED	
18h	SS_SLVOP_BUSY	
19h	SS_INVALID_HOST_SLAVE_OP	
1Ah	SS_GTCB_BEFORE INITIALISE	
1Bh	SS_GTCB_PROTOCOL_ERROR	
1Ch	SS_INVALID_DOORBELL	
1Dh	SS_DNLD_FLASH_FAILURE	Flash memory failed during Download
1Eh	SS_INVALID_OT_DONE	
1Fh	SS_STORAGE	Watch-dog failed to get storage before timeout
20h	SS_VSC	TransferToHost Transaction timeout
21h	SS_POST2A_FAIL	POST2 error
22h	SS_TIMEOUT	Timeout on Transaction from host (> 2 minutes)
23h	SS_SENSE	Additional trace information is available in the Diagnostic Area and a dump may have been saved on disk.
24h	SS_THIRD_PARTY_RESET	SSA Absolute Reset was received from another adapter
25h	SS_LINK_CONFIG_FAILED	SSA link configuration has failed
26h	SS_HEARTBEAT	A Heart-beat reply was delayed too long
27h	SS_FIBREOVERRUN	A Fiber kept control too long
28h	-	Reserved
29h	SS_NWAY_SHOWSTOP	A show-stop message was received from another adapter
2Ah	SS_LRC_FAILURE	The adapter caused a RAID-5 LRC error
2Bh	SS_SHOWSTOP_INTERRUPT	The host software requested a show stop
2Ch	SS_CODE_UNPACK_FAILED	The checksum was bad after unpacking the functional code
2Dh	SS_FAILED_SEQ_NO_ERROR	
40h	XER_NoPrecedingReadyTest	All execute I/O operations must be preceded by a Ready test after a reset or power on
41h	XER_M1DiskGTResourceCount	Disk field is too large when Mode field is 1b
42h	XER_IpnBadResult	IPN transaction failed

Table 50. Adapter Error Types (continued)

Code	Error Type	Description
43h	XER_M0ResourceNotInList	Resource not available when Mode field is 0b
44h	XER_ResourceNotRecognised	
45h	XER_DevNoLongerAccessible	Resource no longer accessible
46h	XER_ReadwriteFailed	
47h	XER_OpenFailed	
81h	TC_BadSYCode	Bad SY_Code
82h	TC_DpbNumberTooHigh	DPB_Number too high
83h	TC_IllegalDPPend	Illegal DP_Pending
84h	TC_IllegalDPAbo	Directive returned DP_Aborting
85h	TC_CalledWrong	Knl_CompleteDirective instead of Knl_CompleteMcbDirective
86h	TC_SafetyCheck	DPB has wrong Safety value in Knl_CompleteDirective
87h	TC_NotPendorAbo	Idsb ≠DP_Pending or DP_Aborting in Knl_CompleteDirective
88h	TC_DenDefault	Den_Default called
89h	TC_DabDefault	Dab_Default called
8Ah	TC_BadPeriod	Bad period in Den_NoOperation
8Bh	TC_BadNOPFlag	Bad flag value in Dab_NoOperation
8Ch	TC_PendInAbo	DP_Pending in Den_AbortDirective2
8Dh	TC_AboInAbo	DP_Aborting returned from abort routine
8Eh	TC_DefaultIsr	DefaultIsr called
8Fh	TC_IntTooBig	Number too large in Knl_ProcessInterrupt
90h	TC_ZeroTime	Zero time in clock block
91h	TC_CTaskZero	Thread routine called with CTask=0
92h	TC_RRError	Knl_RoundRobin called with F/G task
93h	TC_BadExit	DC_ExitThread without SY_None
94h	TC_NoThreads	Thread routine called without thread support
95h	TC_BadAlloc	Den_AllocateBytes > D_MAXMALLOC
96h	TC_MallocFailure	Malloc failure
97h	TC_BadDNode	Corrupt MCB_DestinationNode in Dab_StartTransaction
98h	TC_NotMCB	Knl_CompleteMcbDirective not called with a MCB
99h	TC_SopTcbZero	sop->SOP_Tcb == 0 in DC_SlavdOperation
9Ah	TC_SlaveEPZero	TCB_SlaveOperationEntryPoint == 0 in Den_SlaveOperation
9Bh	TC_NoDefaultService	No default service
9Ch	TC_StrangeAlloc	Strange error from DC_AllocateBytes

Table 50. Adapter Error Types (continued)

Code	Error Type	Description
9Dh	TC_StrangeFree	Strange error from DC_FreeBytes
9Eh	TC_DirectiveFailure	Directive failure
9Fh	TC_Lstqh0	Lst function: qh == 0
A0h	TC_Lstve0	Lst function: ve == 0
A1h	TC_Queqh0	Que function: qh == 0
A2h	TC_Queve0	Que function: ve == 0
A3h	TC_NoBGThreads	No Background Thread support
A4h	TC_TcbNotActive	TCB not active in ipng00
A5h	TC_BadG00Parameter	Bad OT_code in ipng00
A6h	TC_NotVAddress	Invalid DDR for Ddr_VirtualAddress
A7h	TC_SopLengthError	Length truncated in Slave Operation
A8h	TC_IntsDisabled	Interrupts were disabled in Knl_QueueFiber
A9h	TC_IntsEnabled	Interrupts were enabled in KnlQueueFiberIsr
AAh	TC_BadXpnEntry	Xpn_EnterFromUserMode with Intct ≠ -1
ABh	TC_BadXpnExit	Xpn_ExitToUserMode with Intct ≠ 0
ACh	TC_BadCopyDdrToDdr	Bad Knl_CopyDdrToDdr result
ADh	TC_MultipleQueue	Queuing same fiber again
AEh	TC_InterruptsAreOn	Interrupts on when should be off
AFh	TC_InterruptsAreOff	Interrupts off when should be on
B0h	TC_BadSynchro	Bad Synchro
B1h	TC_ListTwice	Item put in list twice
B2h	TC_MemNotReserved	Can’t unreserve memory not reserved
B3h	TC_UnexAdd	Unexpected CC_Add, resource already known
B4h	TC_UnexSetOnline	Unexpected CC_SetOnline, resource unknown
B5h	TC_UnexSetOnline2	Unexpected CC_SetOnline, resource not offline
B6h	TC_UnexSetOffline	Unexpected CC_SetOffline, resource unknown
B7h	TC_UnexSetOffline2	Unexpected CC_SetOffline, resource not online
B8h	TC_UnexRemove	Unexpected CC_Remove, resource unknown
B9h	TC_UnexRemove2	Unexpected CC_Remove, resource not offline
BAh	TC_BadFree	Bad free
BBh	TC_MemCorrupt	Memory corrupt
BCh	TC_UserDefined	User defined

Doorbell Register

This register is used by the host software to signal a microcode interrupt to the adapter.

PCI memory address	BAR_2 + 1C0h (Read/Set)
	BAR_2 + 1C4h (Clear)
PCI I/O address	BAR_0 + 100h + C0h (Read/Set)
	BAR_0 + 100h + C4h (Clear)
Reset value	0000 0000h

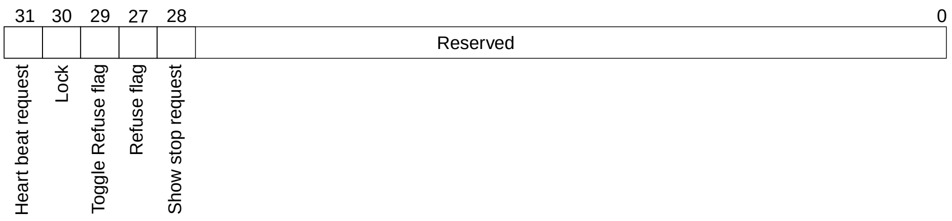


Figure 25. Doorbell Register

Table 51. Doorbell Register

Bits	Name	Description	Type
31	HBreq	**Heart Beat request.** The device driver periodically sets this bit to 1b to request a heart beat check. The adapter replies by setting HBrpy to 1b in the PCI Interrupt register.	R/S/C
30	Lock	**Lock.** This bit is used only by the host software to serialize access to the command interface. It is not examined by the adapter and the microcode interrupt is masked off.	R/S/C
29	TogRef	**Toggle Refuse Flag.** The device driver sets this bit to 1b to change the state of the Refuse flag (see below). The adapter replies by resetting TogRef to 0b when it has toggled the refuse flag.	R/S/C

Table 51. Doorbell Register (continued)

Bits	Name	Description	Type
28	Refuse	**Refuse Flag.** This bit is written by the adapter and read by the device driver. It is only valid when TogRef = 0b. The microcode interrupt is masked off. • When refuse = 0b the adapter can issue IPN transactions to the host normally • When refuse = 1b the PCI gateway in the adapter fails all current and new transactions initiated by the adapter back to the internal master. A slave operation to the adapter that references a failed transaction is terminated back to the host with DE_Failure and a result length of zero. The host must eventually issue an OT_Done or OT_FastDone slave operation for the corresponding transaction in order to free the adapter's resources. This facility is provided to allow PC device drivers to be able to mask the RRQ valid interrupt and stop processing the RR queue (but leaving the CBIOS command interface unaffected) without the adapter going insane. It is needed for Netware boot.	R/S/C
27	ShowStop	**Show-stop Request.** If the device driver sets this bit to 1b the adapter microcode will show-stop, store an error type in the Adapter Error register and set CatErr to 1b in the Interrupt register.	R/S/C
26–0	-	**Reserved.** These bits signal other microcode interrupts that may be used internally by the adapter. The host software must not write 1b to any bit.	R/S/C

Implementation note: When Refuse = 1b the PCI gateway in the adapter fails to the adapter firmware all adapter initiated transactions that are currently in progress with DE_SuccessUnknown. It also fails new adapter initiated transactions, and those queued in the PCI gateway waiting to start, with DE_TransactionAbandoned. This behaviour avoids a deadlock that could otherwise occur if a service in the adapter serializes all transactions (for example, the registry) and it issues a transaction to the host that is suspended by the Refuse flag while the host issues a second transaction to the same service. Similarly, when the host sets Toggle Refuse to 1b the PCI gateway in the adapter must not wait for outstanding transactions from the adapter to complete before setting Refuse to 1b.

The refuse bit = 1b does not itself disable interrupts from the adapter. What it does mean, though, is that adapter initiated transactions that do require host actions will not cause a timeout or deadlock if they are not actioned within the normally required time and no adverse actions are taken if interrupts for these are disabled.

Interrupt Register

The adapter uses this register to signal interrupts to the host software. If a bit is set to 1b and the corresponding bit in the Interrupt Mask register is set to 0b the adapter asserts PCI NTA#.

PCI memory address	BAR_2 + 1E0h (Read/Set)
	BAR_2 + 1E4h (Clear)
PCI I/O address	BAR_0 + 100h + E0h (Read/Set)
	BAR_0 + 100h + E4h (Clear)
Reset value	0000 0000h

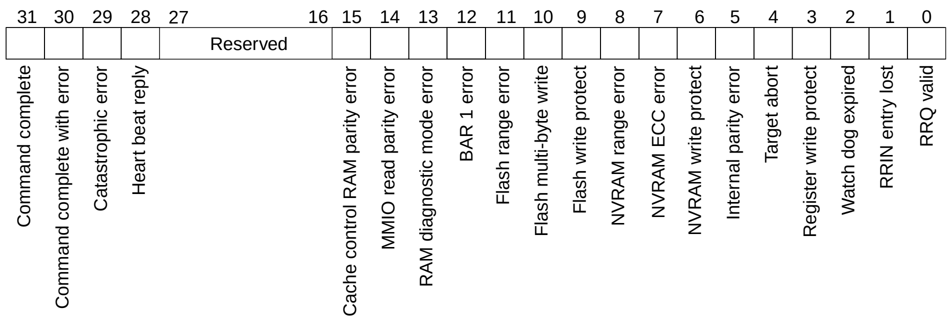


Figure 26. Interrupt Register

Table 52. Local Range Register for Direct Master to PCI

Bits	Name	Description	Type
31	ComDone	**Command complete.** The microcode sets this bit to 1b to inform the host software that the adapter has successfully executed a Command.	R/S/C
30	ComErr	**Command complete with error.** The microcode sets this bit to 1b to inform the host software that a command has failed due to a device or attachment error. The ErrCode field in the Adapter Error register defines the error.	R/S/C
29	CatErr	**Catastrophic error.** The microcode sets this bit to 1b to inform the host software that the adapter has detected a catastrophic error. The Adapter Error register identifies the error. The adapter presents this interrupt continuously until the adapter is reset by host software.	R/S/C
28	HBrpy	**Heart-beat reply.** The microcode sets this bit to 1b to acknowledge that the adapter has received a Heartbeat doorbell from the device driver.	R/S/C
27–16	-	**Reserved.**	R/S/C

Table 52. Local Range Register for Direct Master to PCI (continued)

Bits	Name	Description	Type
15	CCRAMerr	**Cache Control RAM parity error.** The adapter sets this bit to 1b if it detects a parity error when reading the internal cache control RAM.	R/C
14	MMIOerr	**MMIO read parity error.** The adapter sets this bit to 1b if there is a parity error during a host MMIO read of the adapter registers, RAM, NVRAM, or Flash memory.	R/C
13	RAMerr	**RAM diagnostic mode error.** The adapter sets this bit to 1b if the host attempts to access the adapter internal RAM without first setting Diagnostic Access Mode.	R/C
12	BAR1Err	**BAR 1 access error.** The adapter sets this bit to 1b if there is an error when the host accesses SDRAM.	R/C
11	FlashRng	**Flash range.** The adapter sets this bit to 1b if the host software attempts to access a location above the valid range for Flash memory.	R/C
10	FlashMBW	**Flash multi-byte write.** The adapter sets this bit to 1b if the host software attempts to write more than one byte at a time to Flash memory.	R/C
9	FlashWP	**Flash write protect.** The adapter sets this bit to 1b if the host software attempts to write to Flash memory.	R/C
8	NVRAMrng	**NVRAM range error.** The adapter sets this bit to 1b if the host software attempts to access a location above the valid range for NVRAM.	R/C
7	NVRAMECC	**NVRAM ECC error.** The adapter sets this bit to 1b if the host software attempts to read 1, 2, or 3 bytes from NVRAM.	R/C
6	NVRAMwp	**NVRAM write protect.** The adapter sets this bit to 1b if the host software attempts to write to NVRAM.	R/C
5	ParErr	**Internal parity error on write.** The adapter sets this bit to 1b if it detects an internal parity error when the host software is writing to a register, NVRAM, or Flash memory.	R/C
4	Tabort	**Target Abort.** The adapter sets this bit to 1b if it signals Target Abort because of a PCI address parity error, a write of less than 4 bytes to a register, or a write to a protected register.	R/C
3	RegWP	**Register write protect.** The adapter sets this bit to 1b when the host software attempts to write to a protected register or on-chip RAM.	R/C
2	WatchDog	**Watch-dog expired.** The adapter sets this bit to 1b when the internal Watch-dog timer expires because the microprocessor is hung.	R/C
1	RRINlost	**RRIN entry lost.** The adapter sets this bit to 1b when it discards a write to the RRIN register because the RRIN Queue is full and RetryRRIN = 0b in the PCI Control register.	R/S/C

Table 52. Local Range Register for Direct Master to PCI (continued)

Bits	Name	Description	Type
0	RRQval	**RR Queue valid.** The adapter sets this bit to 1b to inform the device driver that it has added an element to the Request/response queue.	R/C

Interrupt Mask Register

This register is used to mask out PCI interrupts.

PCI memory address	BAR_2 + 1E8h (Read/Set)
	BAR_2 + 1ECh (Clear)
PCI I/O address	BAR_0 + 100h + E8h (Read/Set)
	BAR_0 + 100h + ECh (Clear)
Reset value	FFFF FFFFh

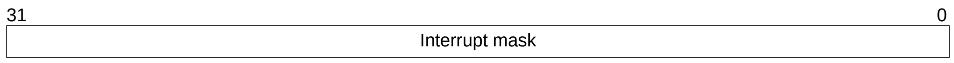


Figure 27. Interrupt Mask Register

Table 53. Interrupt Mask Register

Bits	Name	Description	Type
31–0	Mask	Setting a bit to 1b prevents the adapter from asserting PCI INTA# when the corresponding bit is set to 1b in the Interrupt register.	R/S/C

PIO Error Address Register

This register latches the PCI address of the first PIO error with the adapter as the Target.

PCI memory address	BAR_2 + 1F0h (Read only)
PCI I/O address	BAR_0 + 100h + F0h (Read only)
Reset value	0000 0000h

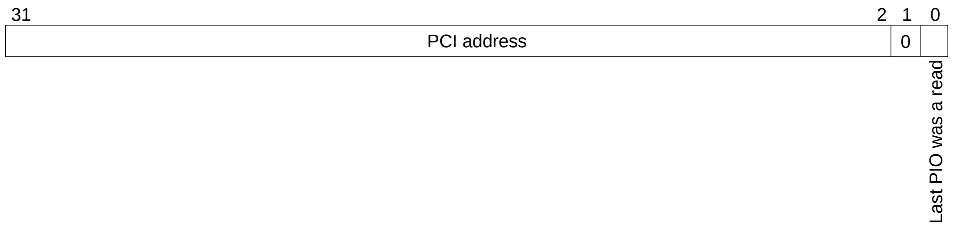


Figure 28. PIO Error Address Register

Table 54. PIO Error Address Register

Bits	Name	Description	Type
31–2	ErrAddr	**Error Address.** When a PIO error is set in the PCI Interrupt register this latch freezes to retain the PCI address of the PIO access that failed	R
1	-	**Reserved.** This bit is always set to 0b.	R
0	Read	**Read Access:** • When Read = 1b the failed PIO was a read • When Read = 0b the failed PIO was a write	R

Chapter 4. Adapter-to-Device Interface

SSA

The Advanced SerialRAID Adapter card has 4 SSA ports that always operate as 2 dual-port nodes. (The adapter does not operate as an SSA switch or as single-port nodes.) Each dual-port node is an Initiator with its own SSA Unique_ID. These differ only in the low-order bit.

Each dual-port can operate at 20 or 40 MB/s and sustain 15 concurrent data transfers. The initiators always use the shortest available path to the addressed node. All messages and data relating to a particular SCSI command use the same path.

The Advanced SerialRAID Adapter supports string and loop networks. The following general restrictions are imposed to ensure good performance:

- To be fault tolerant and allow concurrent maintenance, all networks should be installed as loops rather than strings.
- The maximum number of disk drives in a loop is 48.
- When multiple initiators are permitted on a loop, the following restrictions must be observed:
  – No two initiators should be on the same adapter card.
  – No more than two initiators should be on the same host system. (This allows for fail-over between redundant adapters.)

Every device on an SSA loop operates as a 2-port node. This means that if there is a cable or connector failure between two nodes, a port failure at the node or the node is powered off or removed, and communication is no longer possible on that path, the master initiator is informed of the failure by an Async Alert message. The master then reconfigures the network to use the other port of that node and other nodes that may be affected by the failure. The SSA loop configuration then becomes two SSA string configurations after the failure, but access is still possible to all nodes. When operating as strings, the network is exposed to loss of access to some nodes if a second failure occurs; service action is required to restore the configuration to a loop again.

The adapter distinguishes two types of illegal operation:

1. The adapter may discover an illegal network during SSA configuration. In this case it does not add any new nodes to those that were present at the last legal configuration. Consequently if the adapter detects an illegal network at power-on it does not configure any SSA nodes.
2. An ISAL filter may detect an illegal configuration when it initializes its logical disks.

In both cases the adapter issues an IPN transaction to the device driver to log the error.

There are currently 3 versions of the SSA standards:

SSA-IA/95	SSA-IA/95PH and SSA-IA/95SP. These documents were published by the SSA Industry Association as informal open standards. They support 20 MB/s SSA only and are implemented by all previous IBM SSA products.
SSA Version 1	SSA_PH1, SSA_TL1, and SSA-S2P. These documents are now published ANSI standards (X3T10.1 task group). They also support 20 MB/s only.
SSA Version 2	SSA-PH2, SSA-TL2 and SSA-S3P. these documents are second-generation ANSI standards that include 40 MB/s, speed negotiation, and support for SCSI-3.

The Advanced SerialRAID Adapter implements the SSA-IA/95 specifications with the following additions:

1. 40 MB/s electrical specifications, eye diagrams for example, AC-coupled 19 mA driver and near-end termination. See SSA-PH2.
2. Two sense pins in the external connector to determine the type of cable that is attached.
3. Automatic speed negotiation using a new Link Reset frame.
4. A renegotiation process to allow dual-speed nodes to operate reliably with 20 MB/s interconnections. The speed link is renegotiated by sending a new Link Reset frame when the error rate lies outside certain thresholds. See “Speed Negotiation” on page 88
5. 110 µs ACK time-out to support optical links up to 10 kM long. See “Optical Extender” on page 87
6. An optional priority scheme for the port transmitter that prevents deadlock without constraining the routing of frames.
7. Extensions to the Query_port and Query_port_reply messages to report cable type, current port speed, supported speeds, and 3 frame counters for monitoring performance.

8. A new Async_alert for ‘Port operating at slower than optimal speed’.

These additions to SSA-IA/95 are included in SSA-IA/95+ with which the adapter complies.

SSA Cables

The Advanced SerialRAID Adapter supports both 20MB/s and 40 MB/s cables. The 40 MB/s cables differ in three ways:

- The maximum skew between conductors in the same pair is lower.
- The jacket is colored blue. (A 20 MB/s cable has a black jacket.)
- Pin 1 is strapped to logic ground in each Micro-D connector. The Advanced SerialRAID Adapter senses pin 1 to report whether a 40 MB/s cable is attached.

For lengths up to 20 M the cables have 28 AWG conductors. 25 M cables have 26 AWG conductors.

The use of a 20 MB/s cable does not preclude link operation at 40 MB/s. The Advanced SerialRAID Adapter automatically determines the operating speed from the capabilities of the remote port and the observed link error rate. It is probable that short 20 MB/s cables will operate satisfactorily at 40 MB/s. Therefore it may not be necessary to replace the cables if upgrading from a 20 MB/s adapter to the Advanced SerialRAID Adapter.

Optical Extender

The Advanced SerialRAID Adapter supports the Fibre-Optic Extender 160 that allows a maximum distance of 3 kM per link with multi-mode fiber and 10kM with single-mode fiber. Each port always provides an ACK time-out of 110 µs to permit operation at the maximum distance of 10 kM. The peak data rate is 20 or 40 MB/s full-duplex, depending on the capabilities of the remote node. There is no degradation in the data transfer rate up to 200 M. There is a gradual reduction in the achievable data rate beyond this distance to about 5MB/s at 2 kM and 1 MB/s at 10 kM.

Master Election

Either or both of the two initiators on an adapter card can function as an SSA Master. When an SSA network contains more than one initiator a single master is elected based firstly on the initiator with the highest value of the Master_priority. If there is then a tie among the initiators with the highest Master_priority the initiator with the highest Unique_ID is the master. (The Unique_ID and Master_priority are returned in the Query_node_reply message that is received from each node when an initiator walks the network to build its configuration table.)

Port Configuration

When it is operating as a master the adapter will issue a Configure_port message to each port in the network:

- The port is allocated a tag, port, and return_path for use by a subsequent Async_alert message.
- The A_quota and B_quota for the SAT algorithm are configured according to the guidelines in clause 8.2 of SSA-IA/95PH.

  Note: The adapter does not support multiple SAT regions; all ports are configured to propagate SAT tokens.
- The routing of user-defined characters through the master node in a loop is blocked to avoid continuous circulation.

  Note: The adapter does not originate or use Spindle-sync characters.

Asynchronous Alerts

When it is operating as a master the adapter is responsible for:

- Propagating an asynchronous alert by sending a Master_alert message to all other initiators.
- Performing a third-party quiesce on behalf of a missing initiator.
- Returning the affected ports to normal mode after a transient unrecoverable link error.

Speed Negotiation

The links in an SSA network can operate at different speeds. Each link independently determines its own operating speed.

New SSA nodes have *Dual-speed* ports; for example, the Advanced SerialRAID Adapter can operate at either 20 or 40 MB/s. During the procedure for beginning communication a dual-speed port automatically negotiates its operating speed with the remote port. This allows backwards compatibility with older nodes that do not support the higher speed.

Device Services Interface (DSI)

DSI is a low-speed serial interface for internal use within an SSA storage enclosure. It links each device to a central controller that provides access to VPD, power and cooling status, and slot identification. The host can retrieve this information through the SSA loop using the SCSI-3 Enclosure Services (SES) command set.

The storage devices merely pass through the SES commands from the SSA loop to the DSI interface for interpretation by the central controller.

SES uses the SCSI Send Diagnostic and Receive Diagnostic Results commands. The host software can use the FN_ISAL_SCSI transaction in the Advanced SerialRAID Adapter to issue these commands to a physical disk. (In a cluster the FN_ISAL_SCSI transaction may initially fail with AE_ReservationConflict or AE_FencedOut. If so the host should retry the transaction with the override flag set.)

The SES data has 2 formats — short and long, according to the amount of information provided.

The 7133 Models D40 and T40 disk enclosures support the SES long format.

Configurations

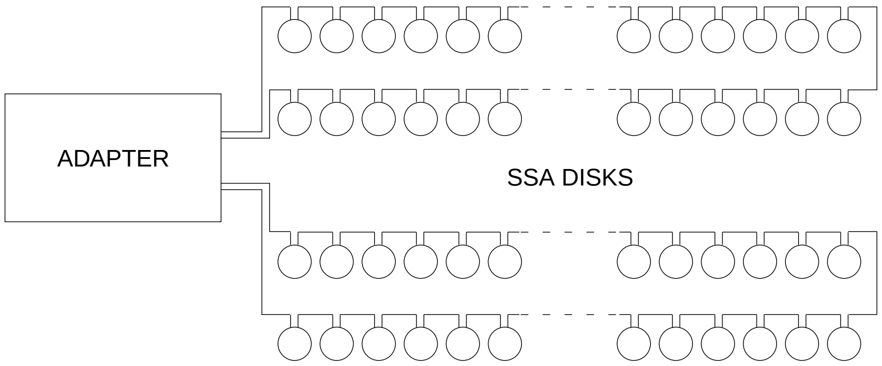
The Advanced SerialRAID Adapter supports the following subsystem configurations:

- Single adapter.
- 2–way cluster.
- N-way cluster for non-RAID only. (3–8 adapters.)

Each configuration has different capabilities and restrictions, as explained in the following sections. The configuration is determined automatically from the SSA network. There is no non-volatile configuration data in the adapter card.

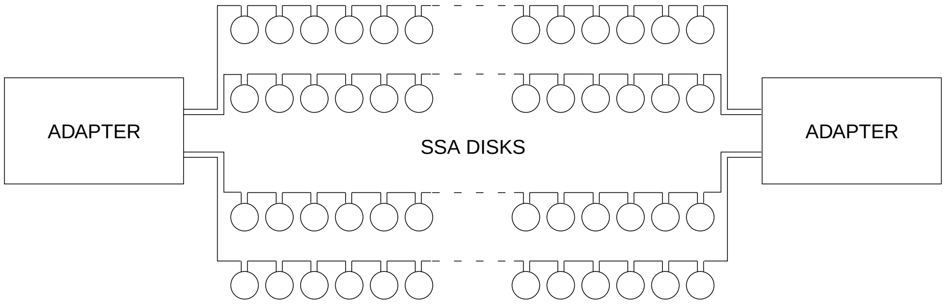
A single host may have multiple Advanced SerialRAID Adapters that attach to independent subsystems.

Single Adapter



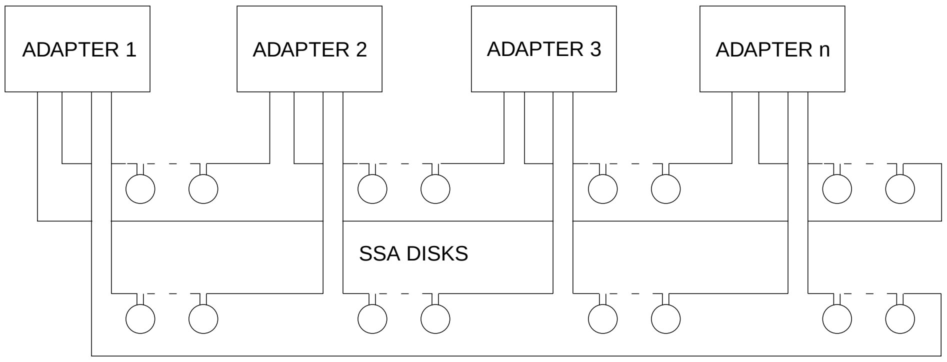
- One Advanced SerialRAID Adapter only.
- 1 or 2 loops with a maximum of 48 disk drives in each loop.
- Mixed non-RAID, RAID-0 and RAID-5 modes, also RAID-1 and RAID-10 if the code level is 50 or higher.
- All members of an array must be on the same loop.
- Read cache is available only for RAID-5 with the integrated cache.
- Write cache is available in all modes.

2–way Cluster



- 2 adapters. (These may be Advanced SerialRAID Adapters, PCI SSA Multi-Initiator/RAID EL Adapters, or Micro Channel SSA Multi-Initiator/RAID EL Adapters for pSeries, RS/6000, or SP/2 systems and may be the Advanced SerialRAID Adapter/X for PC server systems.)
- Both adapters have direct access to any disk drive.
- 1 or 2 loops with a maximum of 48 disk drives and 2 adapters in a loop.
- All members of an array must be in the same loop.
- Mixed non-RAID, RAID-1, RAID-5, and RAID-10 modes. (RAID-1 and RAID-10 are only supported if both adapters are Advanced SerialRAID Adapters with code at level 50 or higher.)
- Read cache is only available for RAID-5 using the integrated cache.
- Write cache is not available.
- No single point of failure.

N-way non-RAID Cluster



This configuration supports an 8-way cluster for non-RAID disks on RS/6000 systems.

- 3, 4, ... 8 adapters. (These may be Advanced SerialRAID Adapters, PCI SSA Multi-Initiator/RAID EL Adapters or Micro Channel SSA Multi-Initiator/RAID EL Adapters for RS/6000 systems. Not supported on PC server systems.)
- All adapters have direct access to any disk.
- 1 or 2 loops with a maximum of 48 disk drives and 8 adapters in a loop.

- Only one of the two initiators on an adapter is permitted in each loop.
- Each loop must not contain more than two adapters that are in the same host.
- Non-RAID only.
- Read cache is not available.
- Write cache is not available.

The number and type of adapters supported with the Advanced SerialRAID Adapters is shown in Table 55.

Table 55. Adapter types in loop

Array Type	**Adapters in loop**	**Adapter types**
Non-RAID	8 (non-PC) 2 (PC servers)	Advanced SerialRAID Adapters PCI SSA Multi-Initiator/RAID EL Adapter Micro Channel SSA Multi-Initiator/RAID EL Adapter
RAID-0	1	Advanced SerialRAID Adapter
RAID-1	2	Advanced SerialRAID Adapters at code level above 50
RAID-5	2	Advanced SerialRAID Adapters PCI SSA Multi-Initiator/RAID EL Adapter Micro Channel SSA Multi-Initiator/RAID EL Adapter
RAID-10	2	Advanced SerialRAID Adapters at code level above 50
Fast Write	1	Advanced SerialRAID Adapters at code level below 50
	2	Advanced SerialRAID Adapters at code level above 50

Adapter Card

The Advanced SerialRAID Adapter is a PCI 5V standard size card. The card measures 312 mm long by 107 mm high, excluding the PCI connector. On the front panel of the adapter and on one of its modules are labels on which is printed the 15-character SSA unique ID of the adapter.

The SSA Fast-Write Cache Option Card is a separate card with a PCMCIA connector that can be plugged in to an Advanced SerialRAID Adapter. The SSA Fast-Write Cache Option Card has 32MB of nonvolatile memory and supporting logic. An SSA Fast-Write Cache Option Card can be removed from a failed adapter and installed on a replacement adapter.

Figure 29 on page 92 shows an Advanced SerialRAID Adapter card. Figure 29 on page 92 shows an Advanced SerialRAID Adapter card with an SSA Fast-Write Cache Option Card installed on it.

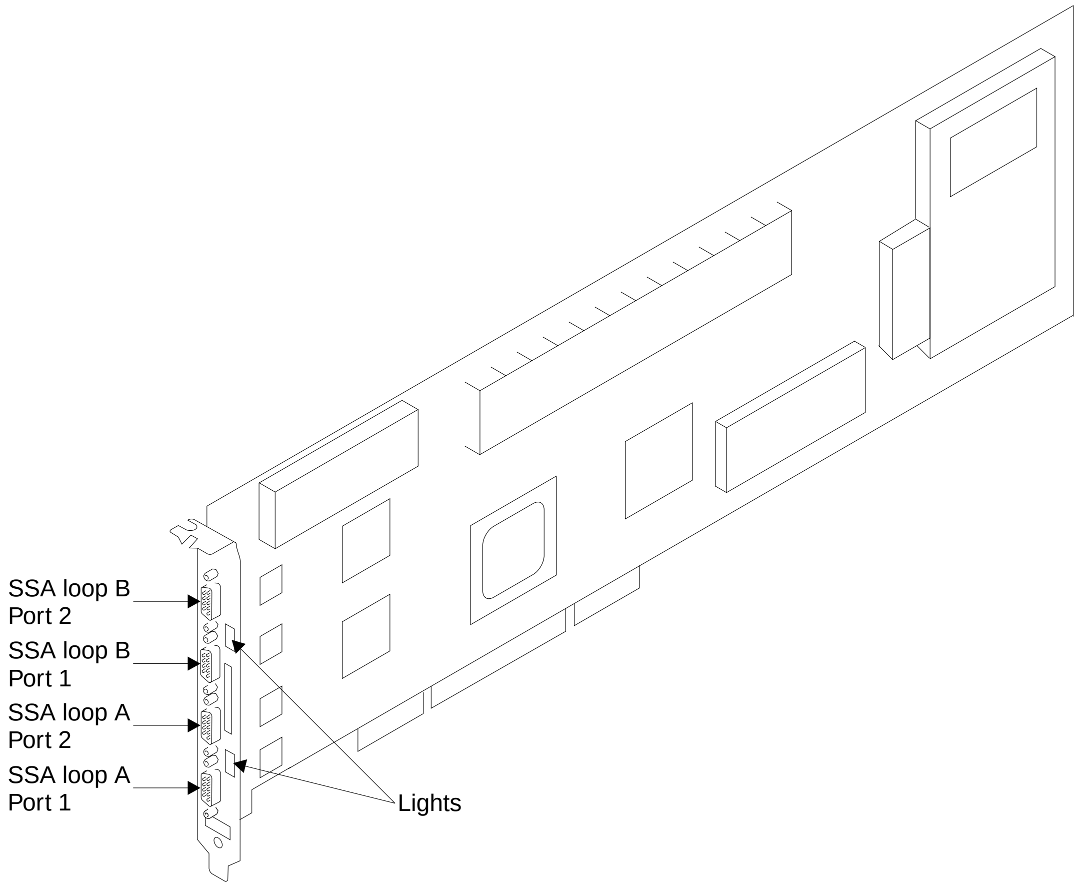


Figure 29. Advanced SerialRAID Adapter Card Layout

SSA Connectors

The adapter card has and 4 external SSA connectors. This allows one of the 2 dual-port SSA nodes to be connected externally to the system unit.

The ports are clearly numbered ‘A1’, ‘A2’, ‘B1’, and ‘B2’ at the connectors. The marking also indicates that ports A1 and A2 are paired, that is, they are connected to the same SSA loop interface chip. Similarly, ports B1 and B2 are paired.

+5 V power is available on the connector to power an external optical extender.

Indicators

A green LED is provided for each Initiator (A and B) to help locate faults in the SSA network:

- When both ports of an Initiator are operational, the corresponding LED is on.
- When one port of an Initiator is not operational, the corresponding LED flashes with a period of approximately 2 seconds.
- When neither port of an Initiator is operational, the corresponding LED is off.
- When the adapter is being identified both LEDs flash together at about 5 flashes per second.

Each LED is mounted between the two external connectors for the corresponding Initiator.

SDRAM Buffer

The adapter has 64 MB of synchronous DRAM that is separately field replaceable. This SDRAM is used for microcode and data caching.

Power Requirements

Voltage	5.0 v ±5%
Current	4.8 A maximum + 0.15 A for each optical extender +0.2 A during cache battery fast charge
Power	24 W maximum
Ripple	100 mV peak-to-peak maximum, dc to 50 KHz

Environment

Operating

Temperature	10 to 60°C at the adapter
Humidity	8 to 80 %, noncondensing
Altitude	0 to 7,000 feet
Cooling	Natural convection
EMC	FCC class A and CISPR 22 class A when packaged in a system unit
Vibration	1G at 6–600 Hz.

Nonoperating

Temperature	−40 to 60°C
Humidity	5 to 80 %, noncondensing
Altitude	−1,000 to 40,000 feet

Chapter 5. Array and Fast Write Filters

The Advanced SerialRAID Adapter provides RAID functions by means of a filter between the device driver and the disk drive. The filter is implemented in microcode that runs in the adapter. The filter presents the image of a single disk drive to the device driver, and uses one or more member members to implement this image. IPN transactions are provided to configure the image. The members are disk drives that are attached to this adapter. All, some, or none of the disk drives attached to an adapter can be members of the filter. There can be up to 16 members in an image. On an

Advanced SerialRAID Adapter, the member disk drives of an array must be in the same SSA loop. This chapter describes the number of members allowed, and the mapping from the image to the members for the filter that is supplied in the adapter.

If the member disks of an array have different sizes, only the capacity of the smallest disk is used for each member.

The adapter can support up to 48 arrays, although their number is limited for certain RAID types because there can be a maximum of 96 disk drives attached.

RAID-0 Filter

RAID-0 does not provide any redundancy and thus it cannot overcome member failures or hardware errors.

The data is striped across the members of the array, evening out skew by spreading the I/O requests evenly across the array. Short transfers usually involve only one member and this enables a high rate of I/O requests. Longer transfers can achieve a higher data rate by accessing several members. In either case there is no write penalty.

The array capacity is the total space available on its members.

Data Mapping

The RAID-0 filter combines N members of equal capacity, that being C blocks on each member. A small amount of space, K blocks, on each member is reserved by the filter for metadata; the rest is used for data storage.

- The number of members, N, must be in the range 2 to 16.
- The strip size, S, can be 8 to 512 blocks (4 KB to 256 KB) in 8 block increments on PC Servers and 32 blocks (16 KB) on pSeries, RS/6000, and SP/2 systems. The default strip size is 16 KB.
- Data is mapped on the disks as follows:

  Blocks 1, 2, ... S of the array are mapped to blocks 1, 2, ... S of the first member.

  Blocks S+1, S+2, ... 2S are mapped to blocks 1,2, ... S of the second member, and so on.

  Blocks NS+1 ... NS+S are mapped to blocks S+1 ... 2S of the first member.

  Blocks NS+S +1 ... NS+2S are mapped to blocks S+1 ... 2S of the second member, and so on.
- The capacity of the array is NS((C-K)//S), where // indicates integer division.

  If the member capacity, C-K, is not a multiple of S, then the excess space is not used.

Algorithms

Read and write data always passes in and out of SDRAM once. The data flow through the buffer is pipelined by the hardware using 2–sided transfers. (Not store and forward.)

For a write transaction the memory is allocated by the cache filter if the data is cached. Otherwise a temporary 8 KB wrapping buffer is allocated by the RAID-0 filter for each member accessed.

For long transfers the RAID-0 filter issues a separate IPN transaction for each strip and the members are accessed in parallel. To avoid losing revolutions on writes at least 2 transactions are issued at a time for each member. (One executing and another queued.)

RAID-0 Array States

A RAID-0 array can be in either of two states:

Good	A RAID-0 array is in the Good state when all the member disk drives of that array are present.
Offline	A RAID-0 array enters Offline state when one or more member disk drives become missing. Read and write operations are not allowed.

RAID-1 Filter

This is supported with code at level 50 or higher.

Characteristics

RAID-1 maintains a mirrored copy of the data on two members of the array. Thus it can overcome a member disk failure or an unrecoverable data error.

- Read performance is enhanced because there is a choice of two disks that can be accessed for the data.

  Write performance is reduced because 2 copies of the data have to be written.
- Copies of the data can be located on disks local to each adapter in a 2–way cluster where the adapters are located remotely from each other by optical links. This permits one of the systems to continue to have access to the array when the remote domain is disconnected or has failed.
- The array capacity is 50% of the total space on the members.
- Rebuild is the process that restores data and parity to a member. It is invoked when a failed member disk is replaced, either manually using the FC_ComponentExchange IACL transaction, or automatically with the hot spare mechanism. Rebuild is also used when an array is created to ensure that both members of a mirrored pair contain the same data

Data Mapping

The RAID-1 filter combines 2 members of capacity C blocks. A small amount of space, K blocks, on each member is reserved by the filter for metadata and the rest is used for data storage.

- There are 2 members.
- The capacity of the array is C-K rounded down to a multiple of 64 KB.

Algorithms

The algorithms used, and the management of the array, are the same as for RAID-10 arrays that are fully described in “RAID-10 Filter” on page 105. The first member of the array is defined as the primary disk and the second member is referred to as the secondary disk. RAID-10 defines the first and third disks to be the primary members. For RAID-1 the first disk is considered equivalant to the first and third disk together; therefore if it is missing it is equivalent to missing the first and third disks of a RAID-10 array. This is required in order to prevent operation on separate member disks of the array when the array becomes split but when separate systems can still access a different member of the array.

RAID-5 Filter

Characteristics

A RAID-5 array stripes data over several members of the array. It maintains a parity strip for each stripe of data. Thus it is able to overcome a member disk failure or an unrecoverable data error.

The data striping evens out skew by spreading the I/O operations evenly across the array members. Short reads usually access only one member, allowing high throughput and a short response time.

Short writes require up to four disk accesses to update the parity, resulting in lower throughput and a longer response time. Therefore it is particularly advantageous to combine fast write with RAID-5. Also, fast write will issue a full stripe write to a RAID-5 array if all the data for the stripe has accumulated in the cache, possibly as a result of several writes to fast write, each of which was for less than a stripe. Writing a full stripe does not incur the overhead of reading the old data and the old parity and so results in improved performance.

Long reads and full stripe writes access several members simultaneously giving a higher data rate.

If one member disk is broken, the array enters a degraded mode. Performance is significantly reduced in degraded mode since all reads and writes to data on the failed member require accesses to all of the other members.

The array capacity is (N-1)/N of the total space on the N members.

Data Mapping

The RAID-5 filter combines N array members of equal capacity, C blocks. A small amount of space, K blocks, on each member is reserved by the filter for metadata; the rest is used for data storage.

- The number of members, N, must be in the range 3 to 16.
- The strip size, S, can be 64 or 128 blocks (32 KB or 64 KB) on pSeries, RS/6000, and SP/2 servers and 32, 64, or 128 blocks on PC servers. The default strip size is 128 blocks (64 KB).
- The stretch size, T, must be 4 or 5 stripes.
- Data is mapped on the disks as follows:

  Blocks 1, 2, ... S of the array are mapped to blocks 1, 2, ... S of the second member.

  Blocks S+1, S+2, ... 2S are mapped to blocks 1,2, ... S of the third member, and so on for N-1 members.

  The exclusive-OR of these strips is written onto blocks 1,2 ... S of the first member. Therefore the parity across all N members is even.

  This pattern is then repeated for a number of stripes, according to the stretch size, T, specified during array configuration.

  The next stretch of ST(N-1) blocks is mapped similarly to the following ST blocks of members 3,4 ... N, 1 and member 2 is used for the parity.

  On the next stretch, member 3 is used for the parity, and so on.

  After STN(N-1) blocks all members have been used for parity and the pattern repeats.
- If the member capacity, C-K, is not a multiple of S, then the excess space is not used and the capacity of the array is S(N-1)((C-K)//S) blocks.
- The number of arrays that can be supported depends on the size of the disk, the strip size, and the code level.

Table 56. *Number of RAID-5 arrays supported with 96 disks*

Disk Size	Strip Size (blocks)	Number of arrays supported		
		Code level < 50	Code level ≥50 (64 MB SDRAM)	Code Level ≥50 (128 MB SDRAM)
4.5 GB	32	20	32	32
	64 / 128	32	32	32
9 GB	32	10	32	32
	64	20	32	32
	128	32	32	32
18 GB	32	5	17	29
	64	10	32	32
	128	19	32	32
36 GB	32	2	8	14
	64	5	17	29
	128	10	32	32

Algorithms

The RAID-5 filter first decomposes each transaction into stripes. Any full stripe writes are dealt with specifically, as described below. The remaining operations are further decomposed into strips or partial strips.

A strip read is normally mapped directly into a read transaction to an individual member. If a strip read encounters a hard medium error or a failed or missing member, then the RAID-5 filter performs a *Reconstruct* to recover the data:

1. Read the corresponding data and parity strips from the other N-1 members into SDRAM. (Some of this data may be found in the integrated read cache.)
2. Calculate the missing strip by performing a multi-way memory-to-memory XOR of corresponding data and parity blocks.
3. Store the XOR result in host memory.

Assuming that the write cache is not being used, a write transaction is handled in one of three ways:

- If a full stripe, that is S(N-1) aligned blocks, is written:
  1. Fetch the stripe into SDRAM from host memory.
  2. Write the new data to (N-1) members.
  3. Calculate the new parity by performing a multi-way memory-to-memory XOR of corresponding strips.
  4. Write the new parity to the remaining member.

  If a member is missing, the array enters the *Degraded* state if it has not already done so. In this case the write to the missing member is simply omitted.

- Each of the remaining strips normally requires up to 4 disk accesses for each member holding data to be written:
  1. Fetch the new data into SDRAM from host memory.

     In parallel, read the old data from disk. (Alternatively this may be found in the integrated read cache.)
  2. Write the new data to disk.
  3. Exclusive-OR the new data with the old data using a 2–way memory-to-memory XOR.
  4. Read the old value of the corresponding parity. (Alternatively this may be found in the integrated read cache.)
  5. Exclusive-OR the old parity with the result of step 3:
     – If the old parity was found in cache then use a 2–way memory-to-memory XOR.
     – Else use XOR as the old parity is read from disk.
  6. Write the new parity from the XOR result in step 5

  A single write operation thus becomes a read and a write to each of two members. Adapters with code at level A000 or above refine the writing algorithm to minimise the number of disk operations when multiple strips, but less than a full stripe, are to be written. Instead of reading the old data from each strip to be written, the firmware reads those strips not to be written and then issues a full stripe write that involves writes to each data and parity member if this involves fewer disk operations than reading the old data.

  If the parity member is missing then the parity write is simply omitted.
- If a data member is missing, the steps to write the strip are:
  1. Fetch the new data into SDRAM from host memory.
  2. Read the old data from the remaining (N-2) members. (Some of this may be found in the integrated read cache.)
  3. Exclusive-OR corresponding blocks of the new data and the result from step 2 using a multi-way memory-to-memory XOR.
  4. Write the new parity from the XOR result.

A write transaction is normally completed when all of the new data has been written to the members. The new parity may be written later. (This is known as *Lazy parity*.) If a data member is missing and the array is in degraded mode then the transaction is not completed until the new parity has been written.

Rebuild is the process by which data and parity is restored to a member. It is invoked when a failed member is replaced, either manually using the FC_ComponentExchange IACL transaction or automatically with the hot spare mechanism. Rebuild is also used to synchronize the parity when an array is first created. It performs the following process for each strip to be rebuilt:

- Read the corresponding strip from each of the other N-1 members into SDRAM. (Some of these may be found in the integrated read cache.)

- Exclusive-OR all of the strips using a multi-way memory-to-memory XOR.
- Write the XOR result to the member being rebuilt.

The rebuild is performed in parallel with any read/write transactions to the array. If there is a single adapter the rebuild process operates on two stripes in parallel so that the source disks do not remain idle during the XOR and write. In a cluster both primary adapters share the work and each adapter operates on one stripe at a time.

The RAID-5 filter issues as many transactions to the members as the available SDRAM allows. A fairness algorithm ensures that memory is shared equitably between all the arrays that are performing I/O.

RAID-5 Array States

A RAID-5 array can be in one of the following states:

Online-Good The array is online and it can be read and written. All the array members are present. All parity data (except that affected by recently completed write operations) is synchronized. No data or parity rebuilding is outstanding. The array is fully protected against the loss of one member.

Online-Exposed

One member is missing from the array. When the array is read, data can be reconstructed for the missing member.

When running without a hot spare, a write can potentially cause data loss if it is interrupted. An attribute that is set when the array is created controls whether writes are allowed in the Online-Exposed state. If writes are allowed, the first write causes the array to enter the Online-Degraded state.

In the Online-Exposed state, the missing member can be reintroduced or replaced. Then, after any necessary rebuilding, the array is returned to the Online-Good state.

Online-Degraded

One member is missing and a write operation has been received for the array. Read and write operations to the array are supported. However, if power is lost before all the parity data has been written, it might not be possible to recreate all the data for the missing member.

The missing member is permanently excluded from the array.

Online-Rebuilding

The array is online and it can be read and written. The full complement of array members are present but data and parity are being rebuilt on one of the members. When rebuilding completes, the array returns to the Online-Good state.

Offline More than one member of the array is missing or has failed. Read and write operations are not supported.

Unknown The array is initially in this state until N-1 members are visible to the filter for the first time.

The movement between states is illustrated in Figure 30.

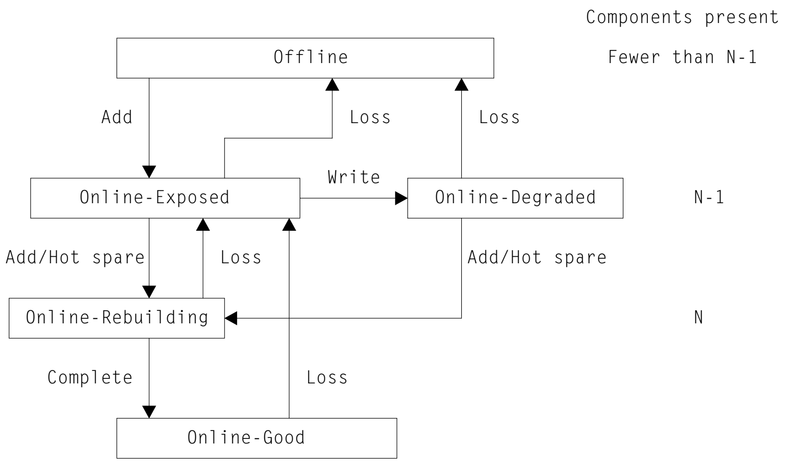


Figure 30. Array State Transitions

Error Recovery

Some disk errors cause a member disk to be deconfigured from an array. When a member disk is deconfigured from an array, it is excluded from the array and never used again for subsequent I/O to that array. Deconfiguring a member disk causes an attempt to replace it with a hot spare if one is available. Introducing a hot spare is not attempted if another member disk is being rebuilt.

- **Parity-in-doubt records.** If a write transaction is interrupted (for example by a power failure) then the parity may not be synchronised with the data. This could subsequently lead to errors in reconstructing other data. To protect against this, the RAID-5 filter makes a record in non-volatile memory of the stripe being updated before writing to the members. This record is normally erased after the write has completed. If the write is interrupted, the RAID-5 filter uses the record to resynchronise the parity when operations resume.
- **Data-In-Doubt records**. Writes to RAID-5 arrays are not atomic and therefore Parity-In-Doubt records are maintained for stripes until all members have been written to ensure that the parity data for those stripes is not used when reconstructing that strip on a replacement disk drive. If, however, the adapter reset that caused the write operation to be terminated was the result of excessive ERPs to a disk drive that subsequently fails, the parity for that stripe is in fact correct and could be used for reconstruction of data for the failed drive. Adapter firmware at level A000 or higher therefore turns off the Parity-In-Doubt flag and sets a Data-In-Doubt flag for a strip that has excessive ERP action (> 2 seconds) when writing to a data component after the parity component has been written. When a Data-In-Doubt flag is set, all other write operations to the array members are suspended until the ERPs are exhausted or the write operation completes. If the write operation does complete,

the Data-In-Doubt flag is reset. If during a rebuild of a disk that has a Data-In-Doubt flag set for a strip, that strip can still be reconstucted using the parity data. Only one member of an array is permitted to have a Data-In-Doubt flag set. Adapter firmware at level A000 or higher also resets the Parity-In-Doubt flag if there is excessive ERP action when reading the data disk, that is before any data or parity has been written to array member disks.

- **Writing in degraded mode.** A RAID-5 array without a spare is particularly at risk when a member is missing and a write is interrupted. In this case the corresponding data on the failed member is lost. (Not just the data that was being written.) For this reason a RAID-5 array may optionally be configured to become read-only when exposed.

  Alternatively, the array may be configured to use a hot spare. If a member fails and a hot spare is available the RAID-5 filter will automatically rebuild the missing strip for the stripe being accessed before performing the write. (No rebuild is necessary for full-stripe writes.)
- **Data scrubbing.** This is a background process that verifies that each data and parity block can be read at least once per week. Any block that cannot be read successfully is reassigned and then rebuilt by reference to the other members of the array.
- **Kill sectors.** A kill sector is a logically-bad block that results from a double error, for example if a medium error is encountered during rebuild. The Advanced SerialRAID Adapter can record the location of up to 128 strips per array, each of which contains one or more kill sectors.

  When a kill sector is encountered during a read, the Advanced SerialRAID Adapter fails the transaction by returning AE_MediumError in the result word and reports the failing LBA in bytes 7 – 4 of the status. All preceding blocks have been transferred. The record of the failure is erased when the bad block is next written.

  The host software can check an array region for kill sectors by issuing an FN_ISAL_Read transaction with FF_Verify set to 1b:
  - If the count specifies one stripe or less, the Advanced SerialRAID Adapter checks that the data can be read from the member disks and then checks that there are no kill sectors. Consequently it will detect medium errors or LRC errors that were previously unknown.
  - If the count field specifies more than one stripe, the Advanced SerialRAID Adapter simply checks that it has already recorded a kill sector. Consequently it will not detect medium errors or LRC errors that have not yet been found.
- **Medium error during a read transaction:**
  1. The LBA of the bad block is recorded by the reassignment manager code (IDSK).
  2. RAID-5 reconstructs the bad block and rewrites it. If reconstruction fails then the block is flagged as a kill sector.
  3. IDSK reassigns the bad block and rewrites it.
  4. RAID-5 completes the read transaction.
- **Member disk hardware error during a read transaction or read phase of a write transaction:**
  1. Reconstruct the requested data for the bad member disk.
  2. Attempt to rewrite the data to the bad member disk. If the rewrite fails because the member is missing, then change the array state to Online-Exposed.

   Otherwise if the rewrite fails then deconfigure the bad member, change the status to Online-Degraded and select a hot spare if possible.
3. Complete the read transaction.

- **Member disk hardware error during the write phase of a write transaction:**
  1. Update the parity data so that it is the correct parity for data that would have been written to the member disk that has failed.
  2. Reconfigure the disk. This is not performed if another disk is rebuilding. In this case:
     – Resync the parity with the old data on the disk
     – Fail the write transaction
  3. Change the array state to Online-Degraded.
  4. Exchange the failed disk with a hot spare if available.
- **Member disk loss prior to a read transaction:**
  1. Select a hot spare if possible. (See “Hot spares” on page 117.)
  2. Reconstruct the requested data for the missing member.
  3. Complete the read transaction.
- **Member disk loss prior to a write transaction:**

  If a hot spare can be selected (see “Hot spares” on page 117.):
  1. Rebuild the stripe currently being accessed.
  2. Perform a normal RAID-5 write.

  If a hot spare cannot be selected and the array is not configured to be read-only when exposed:
  1. Deconfigure the missing member disk, provided that the array will contain no more than 128 strips that contain one or more kill sectors.
  2. Read the corresponding data blocks from the remaining member disks.
  3. Write the new parity from the exclusive-OR of the new data and the data from the remaining members.
- **Member disk loss during a read or write transaction:**
  1. Wait up to 4 seconds for the disk to reappear.
  2. If the disk does not reappear then wait up to 40 seconds for the disk to spin up.
  3. Otherwise deconfigure the missing member disk.
  4. **Read:** Change the array state to Online-Exposed and concurrently reconstruct the requested data from the remaining member disks.

     Write: Change the array state to Online-Degraded. When the state change has been committed, write the requested data in degraded mode.

RAID-10 Filter

RAID-10 is supported by code levels at, or above, level 50.

Characteristics

- A RAID-10 array stripes the data over several members and it maintains a mirrored copy of the data on the same number of members in the mirrored copy. Thus it can overcome a member disk failure or an unrecoverable data error.
- The data striping evens out skew by spreading the I/O operations evenly over the members.

  Short reads usually access only one member, allowing high through-put and a short response time.

  Long transfers may access several members simultaneously giving a higher data rate.

  Read performance is enhanced because there is a choice of two disks that can be accessed for the data.

  Write performance is reduced because 2 copies of the data have to be written.
- Copies of the data can be located on disks local to each adapter in a 2-way cluster where the adapters are located remotely from each other by optical links. This permits one of the systems to continue to have access to the array when the remote domain is disconnected or has failed.
- The array capacity is 50% of the total space on the members.
- Reads to the array involve reads to only one of the mirrored pair of members for each strip concerned. Reads are sent alternately between each of the mirrored members to balance the queues. With code level A000 or higher, successive sequential read commands are directed to the same member rather than alternating between members to improve performance. Also, if one of the mirrored pair of disks is connected to the port through copper cable and the other is connected with an optical fiber cable, data is read from the disk connected with the copper cable.

  If data cannot be read from the chosen disk, the mirrored disk is used to read the required data. Adapter code at level A000 or higher also only reads from the mirrored disk for reads that are required from a disk that is in the process of a microcode download.
- Writes to the array involve writes to each of the mirrored pair of members for the strips concerned. Completion of the Write is not indicated until all members have been written. If a member of the array is missing and the array is in the Online_Degraded state, the performance of writes to strips that would be on the missing disk are actually improved because only a single write is required.

Rebuild is the process by which data and parity is restored to a member. It is invoked when a failed member is replaced, either manually using the FC_ComponentExchange IACL transaction, or automatically with the hot spare mechanism. Rebuild is also used when an array is created to ensure that all members of mirrored pairs contain the same data.

To ensure data integrity for certain failures, some asymetry is introduced to the mirrored pairs (see “Managing Mirrored Pairs” on page 110). The 1st, 3rd, 5th, and so on, disks of an array are considered ‘Primary Member Disks’ and the 2nd, 4th, 6th, and so on, disks of an array are considered ‘Secondary Member Disks’. This property is used when an adapter cannot access the other adapter and only some of the member disks to avoid both adapters using different members independently in this situation.

Data Mapping

The RAID-10 filter combines N members of equal capacity, C blocks. A small amount of space, K blocks, is reserved on each member for metadata by the filter, and the rest is used for data storage.

- The number of members, N, must be in the range 4 to 16.
- The strip size, S, supported for the different platforms can be:
  - 32, 64, or 128 blocks (16 KB, 32 KB, or 64 KB)

    The default strip size is 16KB
- Blocks 1, 2.... S of the array are mapped to blocks 1, 2.... S of the first and second members.

  Blocks S+1, S+2.... 2S are mapped to blocks 1, 2.... S of the third and fourth members, and so on.
- The capacity of the array is NS((C-K) // 2S), where // indicates integer division.

  If the member capacity, C-K, is not a multiple of S, then the excess space is not used.

Algorithms

Reads to the array involve reads to only one of the mirrored pair of members for each strip concerned. Reads are sent alternately between each of the mirrored members to balance the queues.

Writes to the array involve writes to each of the mirrored pair of members for the strips concerned. Completion of the write is not indicated until all members have been written. If a member of the array is missing and the array is in the Online-Degraded state, the performance of writes to strips that would be on the missing disk are actually improved because only a single write is required.

During rebuild of an array, writes to the unbuilt portion of the array cause only the rebuild source to be written as until that portion has been rebuilt it is not used for reading data. Consequently, data written during rebuild is not guaranteed to be protected until the array has been completely rebuilt.

An array member is rebuilt from its mirrored member for the following situations:

- On creation of an array
- After a member has been replaced by a hot spare

Progress indicators of the rebuild are maintained and available to both adapters.

All blocks are read with the verify operation periodically to data scrub the entire array. Any errors are repaired by rewriting the bad block with the data from the mirrored member. The frequency of the data scrub ensures that all the arrays are checked about once a week.

Array States

A RAID-10 array can be in one of the following states:

Online-Good All of the members are present with none deconfigured. The array can be read and written. No rebuilding is outstanding. The array is fully protected against loss of multiple members provided one copy of the mirrored data is still available. There may be some unsynchroised records still being repaired.

Online-Exposed
Some members are missing but have not yet been deconfigured. The array can be read or written, although this will move the array to the Online-Degraded state. The missing members can be re-introduced and the array will then return to the Online-Good state.

Online-Degraded
One or more members are missing or deconfigured and a write has since been received. The array can be read and writtten. The missing members are deconfigured so that they are permanately excluded from the array. If they become available again, they can only be introduced as new members. The state also includes the secondary side operating with the primary side deconfigured, in which case the secondary side maintains knowledge of the members of the primary side to track recovery.

Online-Rebuilding
One or more members are rebuilding. The array can be read and written. When an array is created, it enters the Online-Rebuilding state to synchronise the members. When rebuilding is complete the array returns to the Online-Good state.

If the medium error table fills during a rebuild, the array remains in the Online-Rebuilding state until entries become available in the table.

Offline This can be for any of the following reasons:

- No NVRAM to operate the array.
- Array split across SSA loops.
- The secondary member that holds the configuration sector is present but no primary members that hold configuration sectors are present and the SplitResolution flag is not set.
- Primary members that hold configuration sectors are present but the secondary member that holds the configuration sector is not present and the SplitResolution flag is set.
- Primary and secondary member disks with configuration sectors are present with SplitResolution flag set on the secondary side but the array was not initialised correctly.
- Double failure in a configuration update (configuration sectors, fence sector, label sector, medium error table or unsync table).
- Both of a mirrored pair of member disks are missing, deconfigured, or rebuilding.

Unknown Insufficient array members are present to be able to determine the array configuration, that is, less than two of the first three members. To allow for split arrays to operate, if the secondary disk that holds the configuration sector is available but neither of the primary disks that hold the configuration sectors are available, the array will be in the Offline state unless the SplitResolution flag is set.

Different members in the RAID-10 array may be in different states, for instance one mirrored pair may be rebuilding while another member may be missing and the remaining member of the pair is Online-Degraded. There is a priority of array states of the different members that is used when reporting the state of the array (highest priority first):

1. Unknown
2. Offline
3. Online_Exposed
4. Online_Degraded
5. Online_Rebuilding
6. Online_Good

An error is logged whenever an array state change occurs other than going to Online_Good or Online_Rebuilding.

Error Recovery

Some disk errors cause that member disk to be deconfigured from the array. When a member disk is deconfigured from the array, it is excluded from the array and never used again on subsequent I/O to that array. Deconfiguring a member disk causes an attempt to replace that member disk with a hot spare if one is available.

Errors on Read

An error during a read of a member causes the mirrored copy to be read. If this suceeds, the data is returned to the host and a write is sent to the original failing copy. This rewrite causes a reassignment of the failing sector before data is written to a new sector. If the rewrite fails, an attempt is made to deconfigure the member.

If the second read fails a best effort is made to repair and return the data. Non-colliding medium errors are repaired. If one of the blocks cannot be repaired, all the blocks up to the block in error are returned to the host with the bock address in the status data and an AE_MediumError result. The address of the block is added to the Medium Error table. An AE_HardwareError is returned if this table is full. The entries in the Medium Error table identify the blocks of the array that have suffered a data loss.

Errors on Write

An error during a write as part of a normal host I/O causes a deconfigure of the member to be attempted. If the deconfigure is refused, the write is failed.

An error during a write as part of a re-write following a read error in a host I/O or data scrub operation causes a deconfigure of the member to be attempted. If the deconfigure is refused the operation still continues.

An error during a write as part of a rebuild operation causes a deconfigure of the member.

Deconfiguration of a member is refused if the mirrored member is already deconfigured or is rebuilding.

Medium Error table

This is a list of the addresses of all blocks that have suffered a data loss. This can be due to a medium error for that LBA on both members or a medium error detected when one member is being rebuilt that prevents the member being rebuilt correctly. If a read accesses an LBA that is in the Medium Error table, the read transaction is failed with an AE_MediumError result and no data is returned for that LBA.

Loss of Power

Completion of a write transaction is not returned to the host until either:

1. If the array is configured with fast write, the data has been stored to the local SDRAM and non-volatile cache (and the SDRAM on the other adapter in a 2–way configuration). At some later time, the cached data is destaged to the array member disks. If power is lost during this destage write, completion of the writes will not be returned to the fast write filter.
2. If the array is not configured with fast write, the data has been written to all member disks.

If power is lost before all the writes to member disks completes, data on both mirrored pairs of disks may not be the same. The records being written are marked ’unsync’ records on the meta-data of the array to ensure that if these records are read again in future before a subsequent re-write, the same data is returned from either member (see “Initialisation” on page 113).

Managing Mirrored Pairs

Configuration information of the array is held in a reserved area sector on each of the first 3 member disks of the array. If less than 2 of these configuration sectors can be read or written the array normally goes offline although there are some exceptions to this rule that are described later.

A significant benefit of using RAID-10 is that the mirrored pairs can be located in different site in different power domains resulting in better availability than a RAID-5 array. If the network is configured such that one system and half the mirrored pairs of disks are in one domain and the other system and the other array member disks are in another domain, any power failure of an entire domain does not prevent operations continuing and data can be accessed by the system in the domain that still has power. The configuration utility used when arrays are created displays the identification of the unit that houses the disk drivers to assist in assigning member disks to separate domains.

However, if both domains in a 2 site configuration are both operational but communication is lost between the sites, care is required to ensure that both systems do not continue operating on their copy of the array independently as this would lead to integrity exposures of the data. To avoid this, asymmetry has been introduced into the mirrored pairs. The 1st, 3rd, 5th, and so on, member disks of the array are considered the primary members and the 2nd, 4th, 6th, and so on, are considered the secondary members. Access to at least one of the primary disks that contain the configuration information is required normally for array operations to continue.

- If there is a network partition, the host that has access to the primary configuration disks continues operation and the host that can only access the secondary configuration disk cannot access the array.
- If the host in the site with the secondary member disks fails, the host that has access to the primary configuration disks continues operation.
- If the host that has access to the primary configuration disks and the primary configuration disks fail, that is, only the site with the secondary disks remains, access to the array is withheld to the host that can still access the secondary configuration disk. Special user action is required to allow this host to access the array. It is assumed that the application using the RAID-10 array operates in a failover mode. The system should be set up such that the application runs by default on the primary side and special failover operation is required to allow it to run on the secondary side.

Adapter Operation

The SplitResolution resource dependent value (RDV) determines whether the primary or secondary side of the array can operate when not all the configuration disks are available. The normal mode is SplitResolution is off. When Split Resolution is off and the secondary configuration disk is visible but neither of the primary configuration disks are visible to an adapter, the array goes offline to that adapter.

When the SplitResolution RDV is set and the only configuration disk visible is the secondary disk, access to the array is permitted. If the SplitResolution RDV is set and the secondary configuration disk is not visible, the array goes offline.

The SplitResolution RDV affects initialisation of the array:

- If an array appears with the SplitResolution RDV set, only members from the secondary side are accepted. If members from the primary side appear, they are exchanged into the array and a rebuild process begins. The fact that this has happened is recorded for each of the primary disks.
- If all member disks for the primary side appear and all have the rebuild process ongoing or complete, then the SplitResolution state is reset.

This process of automatically starting a rebuild and then clearing the SplitResolution RDV is independent of which configuration disk is first seen on initialisation:

- If the secondary configuration disk is seen first, the array is brought online with all the secondary disks being used. Any disk that appears from the primary side is rebuilt using the above process.
- If the primary configuration disks are seen first (both primary disks must have the SplitResolution RDV set), access to the primary side is not granted and when the secondary side appears the above rebuild process starts.

If the primary side initialises and the array is read or written and the SplitResolution RDV is not set on the primary configuration disks and later the secondary side configuration disk appears which has the SplitResolution RDV set, the array goes offline and an error is logged. The user will have to determine if the correct data is on the secondary or primary side and reinitialise by changing the SplitResolution RDV value on the appropriate disks. This situation will not arise normally when a site that has been powered off, possibly due to a service action, is powered on again. It may occur when that system and disks are powered on again and communication is not possible between the secondary and primary sites.

Therefore if the entire array initialises concurrently such that primary and secondary disks appear in a single configuration cycle the adapter may expose the secondary half of the array and begin rebuilding the primary side. It will never expose the secondary half of the array and then change its view and expose the primary side instead without taking the array offline.

Communication between adapters ensures that they behave consistently. If one adapter has exposed the primary side of the disks and the other adapter has exposed the secondary side, when these two halves are joined both adapters take the array offline.

The user can configure the array to always be able to continue after a power failure of a single power domain if each of the first two primary disks and the first secondary disk are all in separate power domains. In this configuration after the loss of any power domain, a system will continue to see either all the primary configuration disks or one primary and one secondary configuration disk and will therefore continue to access the array without the need to set the SplitResolution RDV flag.

Operation after failures

The array goes into the offline state if, after any failure or change in configuration, it would otherwise be possible for different systems to continue operation on different disks of mirrored pairs and thereby allow data to be incompatible within a mirrored pair. This could involve cabling either intentionally or unintentionally part of a mirrored pair to a different system from the previous one. An adapter management list is kept in the array meta data of the serial numbers of the adapters that control the array.

The array will continue to be operational for the following errors or reconfigurations:

- Change in the managing adapters (gain or loss) when all the primary and secondary configuration disks are working.
- Single primary configuration disk is missing or cannot be read or written but the other primary and secondary configuration disks are working.
- Secondary configuration disk is missing or cannot be read or written but both primary configuration disks are working.
- Both primary configuration disks are missing but the secondary configuration disk is working and all the adapters in the management list are visible. Adapter firmware automatically sets the SplitResolution flag.
- Both primary configuration disks are visible but cannot be read or written and the secondary configuration disk is working. Adapter firmware automatically sets the SplitResolution flag.

The array will go offline for the following errors or reconfiguration:

- Both primary configuration disks are missing and less than all the managing adapters on the list are visible. The SplitResolution flag needs to be set before operations to the array can continue.

The SDS_HotSpareSplits RDV can be used to control if Hot Spares are introduced when exactly half the members of an array are missing. In a split site configuration when one site loses access to the other, it may be desirable in that situation for Hot Spares not to be introduced when half the disks are no longer visible. When SDS_HotSpareSplits is off and all the secondary disks and the other adapter are not visible, Hot Spares are not introduced. If SDS_SplitResolution has been set on (secondary disks only are being used) and SDS_HotSpareSplits is off, Hot Spares are not introduced when all the primary disks and the other adapter are not visible.

Array Management

Clusters and NVRAM

RAID-10 Clusters with up to two adapters are supported with dual-active adapters and shared disk access. These adapters exchange locks to ensure that write operations do not conflict. Locks are maintained in NVRAM for the local and partner adapter. After 120 seconds of no write activity the array is marked as shutdown and then no NVRAM is required to operate the array.

The array is also marked as shutdown by an FN_ISAL_Flush or an FN_ISAL_Close transaction.

Initialisation

If an un-shutdown array is initialised and the adapter has NVRAM records for that array, then these are copied as ′unsync′ records to the array meta-data. These unsync records identify areas of the array for which a write has started but not completed on both mirrored pairs. The system has been told that the write was not successful, so the content of the data cannot be used by the system. What is important, though, is that both of the mirrored pairs report the same data in future for these areas if read before being rewritten. A special form of rebuild begins that copies the data on one of the mirrored pairs to the other for these unsync records. It is not important which of the disks is used as the source for this copy. When an unsync area has been rebuilt the unsync record is removed.

If during the unsync rebuild the read of one disk fails, then a read of the other disk is attempted. If this succeeds, a rewrite is attempted to the first disk. If the reads from both disks fail, then as much data is copied from one to the other as possible and a medium error is recorded in the medium error table for the remainder.

If during the unsync rebuild a rewrite fails, the failing disk is deconfigured and the unsync record is cleared.

If a disk is deconfigured for which unsync records apply, the action taken for these records depends on whether the deconfigured disk was identified to be the source or target of the rebuild data:

- If the deconfigured disk was identified to be the target of the unsync rebuild or it was unimportant if it was the source or target, the unsync records are cleared when the deconfigured disk is replaced by a hot spare.
- If the deconfigured disk was identified to be the source of the unsync rebuild, the unsync LBAs are converted to medium errors.

If a read or write is attempted for an area that still has an unsync record, the unsync record is resolved before the read or write is allowed.

If an un-shutdown array is initialised and the adapter has no NVRAM records for that array, then it will attempt to inhibit operation to the array by replying AE_AvoidReadWrite to an ISALMgr_TestOneRescrReady transaction. The assumption is that another adapter will appear with NVRAM entries for that array. If an I/O does arrive, then the adapter is forced to treat the entire array as unsynced, and the entire array is rebuilt.

3-Way Copy

3–Way Copy is supported by adapter code levels A000 or higher. It is not supported at lower levels.

3-Way Copying allows a user to add a third copy to an existing RAID-1 or RAID-10 array. A background process then copies data from the original copies to the new copy in a similar fashion to an array rebuild. During the copy process, read and write I/Os may are still permitted to the array with writes updating all three copies of the data. Once the background copy process has completed all write operations continue to update all three copies of the data. At any time after the copy process has completed, the third copy may be split from the original RAID-1 or RAID-10 array to form an independent RAID copy resource. Scripts are provided to synchronize the data for the array and to flush any fast write cache data before the RAID copy array is uncoupled to ensure there is no data cached in the system or in the fast write cache. Before this RAID copy resource can be used, certain operating system specific metadata needs to be modified to make the second copy non-identical otherwise the operating system would see two identical resources with identical file systems. Scripts are available for this.

At the time the RAID copy is split from the RAID-1 or RAID-10 array, the user is effectively taking a snapshot copy of the resource with any cached data destaged to the array. This snapshot copy would then typically be used to perform a backup or test some new application. Once the user has finished with the RAID copy, the RAID copy resource may be re-attached to the same or another RAID-1 or RAID-10 array and the copy process repeated, or the RAID copy array resource may be deleted.

The user is able to user either smit panels or ssaraid commands to create a 3-Way Copy. The first step is to create a RAID copy array which will form the third copy. The RAID copy array must have the following characteristics with respect to the RAID-1 or RAID-10 array for which it will a copy:

- same characteristics (for example strip size)
- exactly half the components

- each component have at least the minimum capacity of any member of the array to be copied

Note that for making a third copy of a RAID-1 array, this requires a RAID copy array with one member. If a 3-Way Copy had previously been created and uncoupled from the array, then a suitable copy array may already be available. New configuration commands are provided by the adapter firmware and ssaraid which take a RAID-1 or RAID-10 resource name and return a list of members for creating a RAID copy array for the 3-Way Copy.

The second step is to link the RAID copy array to an existing RAID-1 or RAID-10 array. The RAID copy array must not be a system disk (that is, have an associated hdisk) at this time. Two new configuration commands are provided by the adapter firmware and ssaraid. The first command takes a RAID-1 or RAID-10 resource name and returns a list of RAID copy arrays suitable for creating the 3-Way Copy. The second command allows a RAID copy array to be attached to a RAID-1 or RAID-10 array to form a 3-Way Copy.

When a 3-Way Copy is created, the adapter does the following:

1. The metadata on the RAID-1or RAID-10 array is updated to indicate that the RAID copy array is to be used as a third copy.
2. The metadata on the RAID copy array is updated to indicate that it is part of the RAID-1 or RAID-10 array.
3. The RAID copy array is taken offline and is no longer accessible (that is, the Resource ID is removed and the RAID copy array serial number is no longer reported by the adapter).
4. The RAID-1 or RAID-10 array remains online and reports that is has extra members in response to configuration queries. The extra members are marked as being part of the third copy.
5. A background process is started to copy data from the RAID-1 or RAID-10 array to the RAID copy array. The work of copying the data is shared between adapters in the same manner as rebuild work is shared.

The background copy process works in the same manner as the rebuild process, copying a stretch of data at a time from the RAID-1 or RAID-10 array to the RAID copy array. The same precautions that are used to block writes whilst a stretch is being rebuilt are applied to the copy process to ensure data integrity.

While the copy is in progress, write I/Os to the RAID-1 or RAID-10 array below the water-mark require a 3rd write operation to the RAID copy array that is executed in parallel with the other two write operations. All three write I/Os must complete before the write is completed to the host. If the write is above the water-mark, then the write just writes to the RAID-1 or RAID-10 array and relies on the copy process to update the third copy on the RAID copy array. During the copy process the RAID-1 or RAID-10 array reports the percentage of the data that has been copied.

Once all the data has been copied to the RAID copy array, the user is able to use a command line, smit or an ssaraid command to uncouple the RAID copy array from the

RAID-1 or RAID-10 array. The recommended procedure is to use the scripts provided as this ensures that data cached in the system is synchronized to disk before the uncoupling. When a 3-Way Copy is uncoupled, the adapter does the following:

1. The metadata on the RAID-1 or RAID-10 array is updated to indicate that the RAID copy array is no longer associated with the RAID-1 or RAID-10 array.
2. The metadata on the RAID copy array is updated to indicate that it is no longer part of the RAID-1 or RAID-10 array.
3. The RAID copy array is brought online. The ISAL reserved sectors and label sector are NOT copied from the RAID-10 array to the RAID copy array and consequently the RAID copy array appears as a free resource.
4. The RAID-10 array remains online but no longer reports that it has extra components in response to configuration queries.

The next step is operating system specific and involves modifying the file system metadata on the RAID copy array so that the RAID copy array is not an identical copy of the RAID-1 or RAID-10 array. Both the hdisk pvid and the metadata stored by the LVM need to be modified.

The final step is to convert the RAID copy array into a system disk (that is, create an hdisk for the device) so that the user can access the copy of the data. The user may also choose to enable fast write at this time if desired.

The following rules apply to 3-Way Copies:

1. Multiple RAID copy arrays cannot be linked to the same RAID-1 or RAID-10 array at the same time.
2. A RAID copy array which is linked to a RAID-1 or RAID-10 array cannot itself be copied by an additional RAID copy array
3. A RAID copy array can be added to an existing RAID-1 or RAID-10 array even if that array is exposed, degraded or rebuilding.
4. The extra copy provides no additional data integrity. Data on the third copy is never used to recover data that cannot be read from the original array.
5. The extra copy is not used to improve read performance.
6. When the extra copy is uncoupled to become a RAID copy array resource, it becomes a RAID copy array resource implemented by the RAID-1 or RAID-10 filters provided by adapter code at level A000 or above. This resource is not backwards compatible with firmware at a level less than A000 and can therefore not be accessed by another adapter at a lower level.
7. If a write comes in to an area of the RAID-1 or RAID-10 array that has yet to be copied, there is no write to the third copy. The background copy operation is relied on to update the third copy on the RAID copy array.
8. Write I/Os to an area of the RAID-1 or RAID-10 array that has been copied require a third write operation to the RAID copy array which is executed in parallel with the other two write operations. All three write I/Os must complete before the write is completed to the host.
9. A RAID-1 or RAID-10 array cannot be attached to a RAID-1or RAID-10 array to give four copies

Hot spares

RAID-1, RAID-5, and RAID-10 arrays can be configured to use hot-spare disk drives. If a hot spare disk is available when a member fails, the hot spare is automatically used to replace the failed member. A hot spare is required on each SSA loop in which there are array members.

When a disk drive fails and is missing from an SSA loop, it is replaced by a hot spare when a write transaction is received or, if the array has no incorrect data, when a read transaction is received.

It is recommended that hot spares are available. A write operation to an array that has a member missing causes that array to enter the degraded state. Unless the array is operating in the read-only-while-exposed mode, if an array is in the degraded state and if a write operation to the parity disk has not been completed and if there is a loss of power, data for the unwritten blocks cannot be recreated when the missing disk is replaced.

With code at level 50 or higher, the control of hot spares has been enhanced by being able to assign different hot spares to selected arrays. With this capability, hot spares can be assigned to particular arrays (for example, if the spare is preferred to be in the same package as the array) or in the case of RAID-1 or RAID-10 they can be assigned to particular members (for instance, to keep the spare on the same physical site as the array members it could replace). A hot spare manager controls the hot spare substitution. It also checks that the hot spares are still working at every healthcheck.

The array configuration tools allow each array member and hot spare to be configured with a pool number. When a member is missing or fails, the adapter chooses a hot spare from the specified pool in preference to any other hot spare. All hot spare managers keep a count of the number of hot spares in their assigned pool and this is used to determine how to reconfigure the hot spare configuration following a hot spare takeover. They all also keep a count of the minimum number of hot spares that are permitted in the pool and this is used to set a threshold so that users can be alerted to the need to replace disks before they run out of hot spares. The default is that all hot spares and members are assigned to a pool number of zero which is reserved to mean that a global hot spare policy is to be used.

A flag can be set for arrays and members to control whether a hot spare only from a preferred pool should be used or whether a hot spare from a preferred pool should be considered first but, if one is not available, a hot spare from another pool can be used. If a non-preferred hot spare is used the adapter sends an error log to identify this as the user may need to change the array members for the desireable configuration.

The Advanced SerialRAID Adapter with code lower than level 50 reports an error log every healthcheck for each array that has no suitable hot spares. The Advanced SerialRAID Adapter with code at, or higher than, level 50 replaces these error logs with a single error log that indicates that either a hot spare pool is empty or that a pool has been depleted below the specified minimum number of disks. Error logs are also reported if different hot spare disks in a pool present different views and are then considered to be out of sync, for example because disks have been moved or recabled.

Out-of-order Writes

A resource dependent value (PageAlignedSplits) is defined for RAID-0 and RAID-5 that permits data to be written out of order even when the FF_Split flag is off. This attribute can be set when an array is configured.

An unaligned 4 KB page is the first 4 KB section of an array address space (starting at address 0) and each subsequent contiguous 4 KB section of its address space.

If a write operation whose data lies wholly within an aligned 4 KB page is interrupted and it does not complete, a subsequent re-read of the corresponding data results in either all the old data, or all the new data, or a mixture of old and new data with a single transition from new data to old data at some block boundary within the operation.

An operation that straddles more than one aligned 4 KB page is regarded as having been broken into multiple aligned 4 KB operations and each is treated independently by the above rules. There is no guarantee of the order that data is written in separate 4 KB pages.

When the PageAlignedSplits attribute is on, strips are written in any order. The RAID filter sets the FF_Split flag in the write transaction to each member to match that on the write transaction to the array. Therefore if FF_Split is turned off, data is guaranteed to be written in ascending LBA order on each member. When the PageAlignedSplits attribute is off and FF_Split is off, strip writes take place in order of ascending logical block addresses for the array.

Fast Write

The fast write filter adds fast write caching capability for individual disk drives and arrays. A write operation to a fast write filter results initially in the data being written to both the non-volatile fast write cache and the volatile data buffer on the adapter; after which, status is sent to indicate that the operation is complete. At some later time, the data is written to the underlying disk drive or array; after which, the data in the non-volatile fast write cache is discarded. If a second write operation is addressed to the same location, the later one might replace the earlier in the buffer before it has been written to the underlying member. If multiple writes are done to adjacent locations, they may be combined into a single write to the disk. The service time is much shorter because completion is signalled as soon as the data is in the buffer.

If power fails before the data is written to disk, the data is preserved in the fast write cache. When power is restored the adapter, it writes the data to disk. If the adapter fails, any data not yet written to disk is preserved in the fast write cache, which can be removed and fitted to the replacement adapter card. When this new adapter is powered up, it writes the data to disk.

The fast write cache size is 32 MB. An LRC is generated for each page in the cache for integrity checking after a loss of power.

Data is preserved in the non-volatile cache using an internal battery with a life of at least 7 days on load. When power is restored after a power failure, the adapter destages this data to disk.

Any array or non-RAID disk can be configured for fast write. For these arrays or disks all transactions from the host are sent to the fast write filter.

Data held in the non-volatile cache is protected with an error correcting code. For every 4 words of data, a single word of check data is held for integrity of data and to provide a correction mechanism for certain failures. Data held in the non-volatile cache is periodically checked to verify there has been no failure. If a failure is detected, the data still in the volatile SDRAM is destaged to disk, an error log is generated and the cache is disabled. Future writes to the fast write resources do not write to the non-volatile cache and completion of the write is not returned until the data has been written to the disk or member disks of an array.

The non-volatile write cache is a separate field replaceable unit on the adapter card. When an adapter card is replaced, the non-volatile write cache card is moved from the failing adapter card to the replacement adapter card. This ensures that any data that had not been destaged when the adapter card failed can now be destaged on the replacement adapter card.

The execution of the transactions is as follows:

Write Operations

- If the length of data is less than a defined value (which can be set by the user), the data is saved in the cache and completion is signaled before data is written to disk. Also, the user can specify the range of logical block addresses that are to be candidates for saving in the cache.
- If the length of data is greater than a defined value (which can be set by the user), the transaction is passed to the array or individually-accessed disk and data is written to disk before completion is signaled.

Read Operations

- If all the requested data is held in the cache, the fast write filter sends it from the volatile data buffer to the host without involving the RAID or disk services.
- If none of the requested data is held in the cache, the read transaction is forwarded to the RAID filter or disk service for execution.
- If part of the requested data is held in the cache, this data is destaged to the disks before the read transaction is forwarded to the RAID filter or disk service for execution.

Data is destaged from the non-volatile data buffer to the disks at the following times:

- When the resource is closed.
- When a FN_ISAL_Flush transaction is executed for the resource.
- When the cache contains 70% of its maximum capacity. Data is not destaged immediately to benefit from the possibility of merging writes to disks.

- When data has been in the cache for 2 minutes.
- When a contiguous block of data has been saved in the cache. The amount of contiguous data for the RAID filter is a stripe (Nxstrip).

Clusters

2–way fast write is supported by code at level 50 or higher. In a 2–way cluster the cache filter sends a copy of the write data to the other adapter through one of the device loops. Each fast-write logical disk is owned by one of the adapters which keeps two copies of the write data, one in SDRAM and one in the non-volatile cache card. The other adapter also saves a copy of the data in SDRAM. Thus there are normally 3 copies of each block of write data. A write transaction is only completed when all 3 copies have been created.

If the owning adapter fails, then any data that was in its write cache is still available from the remaining adapter. The remaining adapter also assumes responsibility for destaging the data to disk.

The cache filter supports dual-active adapters but the implementation is not optimised for shared disk access. This means that only one adapter can open each logical disk for read/write at a time. The adapter that first writes to the logical disk becomes the owning adapter. The other adapter can still see the logical disk but cannot write to it. If the non-owning adapter receives a write request to the logical disk, it fetches the data to be written into its SDRAM and sends the write request to the owning adapter. The owning adapter fetches the data from this SDRAM into its SDRAM and non-volatile cache card memory. When it has saved both these copies of the data, the owning adapter informs the other adapter of the completion and that adapter sends the result to its host to complete the write transaction. Only the owning adapter normally later destages the data to the disk.

If the owning adapter fails, the remaining adapter can take over and there is no data loss. (High availability.) If power fails then the owning adapter must return to allow access to the data when power is restored.The mirroring of write data in both adapters also ensures coherency, that is, if a block is written through one adapter and later read through the other then the read returns the correct data.

The identification of the owning adapter is held in a configuration sector of the logical disk and in non-volatile RAM on the adapter. An out-of-sync flag identifies that there is an owning adapter. The adapter that receives the first write to the logical disk becomes the owning adapter. It remains the owning adapter until any of the following:

1. There are no write requests to the adapter for a 5 second period. The owning adapter destages all outstanding data to the logical disk and clears the out-of-sync flag.
2. The write activity from the non-owning adapter exceeds that of the owning adapter by a defined ratio. When this threshold is exceeded, the owning adapter destages outstanding data to the member disk, or disks, and clears the out-of-sync flag. The other adapter is then allowed to write directly to the logical disk and it becomes the new owning adapter.

3. The owning adapter fails or is removed from the network. The other adapter then destages any outstanding data for that logical disk from its SDRAM before clearing the out-of-sync flag and becoming the new owning adapter.

If 2–way fast write is used in a shared disk environment where writes are issued to both adapters, overheads are incurred that have an effect on the performance achievable. The maximum operations per second possible when some writes are sent to the non-owning adapter are less than if all writes were sent to the owning adapter. Also if the rate of sending write requests varies, the owning adapter may be switched and this results in an initial period of longer response times while data is being destaged before the performance improves when the adapter of heaviest use becomes the owning adapter.

The size of the non-volatile cache is 32 MB. If the Advanced SerialRAID Adapter has 64 MB SDRAM and is operating in a 2-way fast write environment, only 16 MB of the non-volatile fast write cache is used on each of the adapters. For pSeries, RS/6000, and SP/2 servers the size of SDRAM can be increased to 128 MB. With 128 MB SDRAM, the full 32 MB of the non-volatile fast write cache on each adapter in a 2–way system is used.

Array Configuration

An array-configuration utility is provided to create, delete, and change an array. Essential array information is maintained in the reserved area of the member disks of each array. The information depends on the type of RAID array.

RAID-0 information includes:

- Array serial number
- Member serial number
- Resource dependent values

RAID-1 information includes:

- Array serial number
- Member serial number
- Resource dependent values
- SplitResolution flag

RAID-5 information includes:

- Array serial number
- Member serial number
- Resource dependent values
- Bit maps (for bad parity)
- Out-of-sync flag

RAID-10 information includes:

- Array serial number

- Member serial number
- Resource dependent values
- Strip size
- SplitResolution flag

All the essential information is on the disks. The bit maps may be supplemented by information in the NVRAM. The information is stored in such a way that for RAID-0 arrays:

- If any disk is removed it is possible to identify the serial number of the array.

For RAID-1, RAID-5, and RAID-10 arrays:

- If any disk is removed, it is still possible to operate the array.
- If more than one disk is removed, it is possible to identify the serial number of the array but, possibly, no more.
- If any update to the information is interrupted or fails for any reason, it is still possible to determine the state of the array.

Clusters

The Advanced SerialRAID Adapter adapter allows 2 host systems to share RAID-1, RAID-5, and RAID-10 arrays. These hosts may also share devices that are not configured in arrays; in this case up to 8 adapters can be attached to the SSA loop for non-PC systems and up to 2 adapters for PC systems.

Each system can execute to shared arrays all the functions currently provided for unshared arrays plus additional locking and fencing functions to allow systems to control access to the arrays.

No single failure of any member in the system can cause an array to be inaccessible to all healthy systems.

Member disks of an array must all be on the same SSA loop. All adapters on the same loop as a Advanced SerialRAID Adapter where any disk is configured for RAID, must have a SSA upper level protocol = FCh and a Query_protocol list = 02h. This identifies that the Advanced SerialRAID Adapter supports RAID from two adapters. Up to two adapters can control the arrays that exist on two SSA loops between both adapters.

The two adapters act as peers for operations to arrays that have been configured from disks on the SSA loops. Each can read and write directly to the disks, but need to request locks to the other adapter to ensure their operations do not conflict. Some operations do not require a lock from the partner adapter before execution, for example a read if the partner has not locked the area of LBAs. All write operations require a lock with the partner adapter. A synchronizing agent controls the transfer and acknowledgment of locks. A write operation involves requesting a lock for the required strips from the partner adapter. The partner grants the lock and remembers the strip that may now contain out of date parity information in its NVRAM. If the first adapter fails before the write completes, the partner has the information in its NVRAM of strips with out of date parity and is able to regenerate the parity after the adapter failure.

By exchanging locks during operations and by keeping synchronizing information on each array, whenever one of the partner adapter fails, the other can continue to operate and there is no impact to any of the arrays

If there is only one adapter in the network, the array filters continue to request locks through the synchronizing agent which immediately grants all lock requests on behalf of the absent adapter.

Fast Write Cache is supported provided there is only one adapter in the SSA loop if the code level is below level 50 but is supported on two adapter clusters if both adapter's code levels are at, or higher than, level 50.

Both adapters in a 2–way configuration may be on the same host system and on differant PCI buses. In this configuration on a pSeries, RS/6000, or SP/2 server, if one adapter fails, the device driver will fail over all outstanding I/O operations from the failed adapter to the remaining adapter with no host involvement. This permits operations to continue and data to still be accesible even when one adapter fails. The adapters should be in separate host systems to permit data to still be available after a failure in the host system other than an adapter failure.

Chapter 6. IPN Transactions

Introduction

The Advanced SerialRAID Adapter provides a registry service, filter services, and an adapter service. Transactions are transmitted across the PCI interface to these services in a Gateway Transaction Control Block (GTCB). The format of the GTCB is defined in “Gateway Transaction Control Block (GTCB)” on page 13.

Services are at the heart of the IPN architecture. They form the server side of the client-server model. All communication to and from a service uses IPN transactions. Each server can be said to exist on a node and have its own unique service number. The combination of the node and service number form the network address of the service.

Generally services are used to gain access to a resource, whose size and importance can vary greatly. Large and complicated database systems can be implemented as an IPN service and therefore can have access to a few kilobytes of nonvolatile RAM. The important attribute of IPN is that the interface to the two above examples are very similar.

Every service has a service language that describes the way that the communication to that service must be performed. IPN Storage Access Language (ISAL) is the language used by the disk service. IPN Array Configuration Language (IACL) is the language used by the array-configuration service.

When a service is installed into an IPN kernel the type of the service must be declared. This effectively declares what type of language the service understands. The service type is a one-byte code and can be one of the following:

TP_ISAL	A disk or other resource that acts like one (see “IPN Storage Access Language (ISAL) Services” on page 159)
TP_Registry	A local information server (see “Registry Service” on page 131)
TP_CfgAgent	An array configurator (see “Array-Configuration Service” on page 227).
TP_AdapterService	An adapter service (see “Adapter Service” on page 204).
TP_ErrorLogger	A service in the device driver that receives error logs.

Every IPN node contains a router service which is service number SN_Router and of type TP_Router. This is for the internal use of IPN and should not have transactions issued against it.

Device Addressing

Logical disks are identified by a resource ID. The host uses this resource ID to open the resource. During the process of opening the resource, a handle is returned for the resource. The host uses this handle when sending transactions to the resource.

Resource ID

The resource ID is an identifier that is passed to the resource manager to identify which logical disk the caller is referring to. The resource ID structure is shown in Figure 31. Byte 3 is the owning-module-type field. This is a number that identifies the logical owner

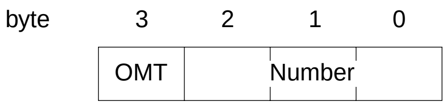


Figure 31. Resource ID

of the resource. An SSA disk might be logically owned by the host disk driver; or, if it is part of a disk array, it might be owned by the RAID-5 manager.

The following values are used:

```
  OMT = 1 - Not Owned by anyone
        2 - Device Driver Physical Adapters
        3 - Device Driver Physical Targets
        4 - Device Driver manually configured logical disks
        5 - Device Driver automatically configured logical disks
        E - Fast write
        F - RAID-0
        G - RAID-1
        K - RAID-5
        O - RAID-10
        T - 3rd Copy
        W - Disowned
        X - Nvram entry
        Y - Hot spare disk
```

The lower 24 bits of the resource ID is a number that is used to identify which resource is being used. This number may be set by the ’user’ or it may be assigned automatically by the resource manager.

All but one of the OMT values are set automatically. This is done by asking the registry for a temporary resource ID (using the FN_REGY_GetTempResrcID transaction). The result of this is a unique 24-bit number that when added to the OMT will form the resource ID.

The exception is OM_DriverManualDisk, which is used in a similar way to a SCSI target number in the system and is permanently assigned to a disk. This information is kept in the device label record.

ISAL Reserved Area

ISAL disk resources maintain a reserved area of 512 byte blocks. The number of blocks available is reported in the FN_ISALMgr_Characteristics transaction. Each block can be held on a sector that is formatted for more than 512 bytes, but the padding is stripped from the extra bytes by the disk service. The SSA Disk ISAL manager internally has 32 blocks mirrored of which 29 are available to the using filter (one is used for a label block, another for a fence sector block, and another for the IDISK control block). The blocks are normally mirrored on a disk so that 64 sectors are required. Each cascaded filter can use some of these reserved area blocks and make the residue available to the next filter. The normal ISAL interface (FN_ISAL_Read/Write) is used to read and write this area. A flag specifies that the I/O should be directed to the reserved area. There are a number of restrictions that apply to data in this area which are:

- I/O operations can only be one block in length.
- The first 16 bytes of all blocks are reserved, so each block must have the format shown in Figure 32.
  The signature is a unique 8-byte field that is used to identify the sector as containing

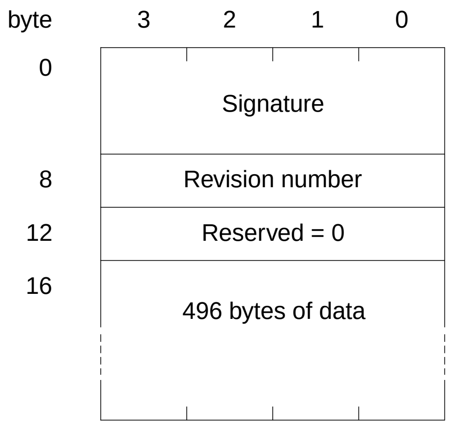


Figure 32. ISAL Reserved Area Sector Format

valid reserved area data (all the 32 sectors share the same signature). The signature field is an ASCII string ‘ISALSIGN’. The revision number is used when reading the reserved data. The resource manager should read both mirrored copies and return the sector that contains the highest revision number (normally both are the same). Then follows a 4 byte reserved field that must be zero, and then 496 bytes of user data.

When writing a sector in the reserved area, the call sets up the first 16 bytes according to the rules here, and it is recommended that the new revision number is higher than the old value.

In the case of a disk, the reserved area starts 128 sectors from the end of the disk. The first sector of the reserved data (sector 0) is reserved as the device label record. Sector 1 is used for the fence sector. Sector 2 is used by IDISK. Sectors 3 to 31 appear as ISAL reserved blocks 0 to 28. In addition to these blocks are 64 blocks used by IDISK to hold data blocks that have been reallocated by IDISK because they could not be reassigned.

Label Record

The label record is where the owning-module type (OMT) is recorded. If the OMT is OM_DriverManualDisk then another number is also stored, called the ’disk number’. For this Owning Module Type, the resource ID consists of the OMT in byte 3 (msb), zero in byte 2, ’disknumber’ in byte 1, 0 (lsb). When the ISAL resource manager reads the label record it should only look at the disk number if the OMT is OM-DriverManualDisk. The label record is kept in the ISAL reserved area but is not accessible by a read or write. It is only written when an OMT other than OM_DriverAutomaticDisk is set.

The label record contains:

- Signature (8 bytes)
- Revision number (4 bytes)
- OMT (1 byte)

Registry Service

The function of the registry service is to maintain a database of IPN information. Each node runs a copy of the registry service. The registry service has a fixed service number (0000 0001h).

The registry service keeps a list of all of the services running on its node, and also a list of all the other nodes that can be accessed through a gateway from its node. Using these two lists, it is possible to walk the whole IPN network and discover what services are available.

In addition, the registry service performs a number of asynchronous notification services, such as error logging. The error logging process registers itself with all the registries. When a module detects an error, it reports this to its local registry service. The registry service sees that the error is sent to the error logger. This approach avoids the error logger having to register itself with every module that is capable of logging an error.

The registry service supports the following application transactions:

Table 57. Registry Transactions

Transaction	**Minor_function**
FN_REGY_SystemVersionInfo	10
FN_REGY_GatewayNodeList	11
FN_REGY_Servicelist	13
FN_REGY_ConnectForNodeChange	14
FN_REGY_DiscForNodeChange	15
FN_REGY_NodeChangeToRegistry	16

Table 57. Registry Transactions (continued)

Transaction	Minor_function
FN_REGY_NodeChangeFromRegistry	17
FN_REGY_ConnectForErrorLogging	18
FN_REGY_DiscForErrorLogging	19
FN_REGY_LogErrorTo Registry	20
FN_REGY_LogErrorFromRegistry	21
FN_REGY_ConnectForResrcChange	22
FN_REGY_DiscForResrcChange	23
FN_REGY_ResrcChangeToRegistry	24
FN_REGY_ResrcChangeFromRegistry	25
FN_REGY_ResrcList	26
FN_REGY_GetTempResrcID	27
FN_REGY_ConnectForHealthCheck	28
FN_REGY_DiscForHealthCheck	29
FN_REGY_HealthCheckToRegistry	30
FN_REGY_HealthCheckFromRegistry	31
FN_REGY_SerialNumberSearch	32
FN_REGY_TestResrcsReady	33
FN_REGY_SetClusterNumber	34
FN_REGY_TestOneResrcReady	35
FN_REGY_SyncHCheckToRegy	36
FN_REGY_SyncHCheckFromRegy	37

FN_REGY_SystemVersionInfo

This transaction can be sent to a registry service to obtain its code level.

Minor_function 10

Parameter_DDR
Null

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to the buffer allocated to receive the following data:

Byte	3	2	1	0
0	Version			

Version This field contains a 32-bit unsigned integer that identifies the current level of the registry code.

Result The following result fields can be returned:

AS_Success

FN_REGY_GatewayNodeList

This transaction returns the numbers of all the IPN nodes that might be known to the system. Further investigation is required to determine if a node is currently attached. The adapter registry services return a list of all nodes that could be connected for this configuration.

Minor_function 11

Parameter_DDR

Null

Transmit_DDR Null

Receive_DDR This is a pointer to a buffer which will receive the following data:

Byte	3	2	1	0
0	Node			
4	Node			
.				
n	Node			

Node This field contains the number of an IPN node that might be attached.

Status_DDR This is a pointer to a buffer that receives the following data:

Byte	3	2	1	0
0	Count			

Count This field contains the number of entries in the received data DDR.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_REGY_ServiceList

This transaction returns the numbers of the services that are running on the same node as the registry service.

Minor_function 13

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0			Type

Type This identifies the type of services that should be reported. “Service / Transaction Directives” on page 325 gives the full list of types supported; these include:

TP_ISAL	Disk service (or something that acts like one)
TP_Registry	Local information server
TP_AdapterService	Adapter service
TP_CfgAgent	Array-configuration service

Transmit_DDR Null

Receive_DDR This is a pointer to a buffer that receives the following data:

Byte	3	2	1	0
0	Service			
4	Service			
.				
n	Service			

Service This identifies the services of the requested type on this node.

Status_DDR This is a pointer to a buffer that receives the following data:

Byte	3	2	1	0
0	Count			

Count This field contains the number of entries in the received data DDR.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_REGY_ConnectForNodeChange

This transaction registers the caller as being interested in node-change asynchronous alerts.

Minor_function 14

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Node			
4	Service			
8	Reserved= 0			Synchro

Node This identifies the IPN node to which node-change asynchronous alerts should be reported.

Service This identifies the service of the IPN node to which node-change asynchronous alerts should be reported.

Synchro When the synchro field is SR_Synchro, the registry service sends node-change asynchronous alerts for all nodes known to the registry service before this transaction completes. When the synchro field is SR_NoSynchro, node-change asynchronous alerts are only sent for nodes that register after the transaction.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_TableFull

FN_REGY_DiscForNodeChange

This transaction registers the caller as being no longer interested in node-change asynchronous alerts.

Minor_function 15

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Node			
4	Service			
8	Reserved = 0			Synchro

Node This identifies the IPN node to which node-change asynchronous alerts had been reported.

Service	This identifies the service of the IPN node to which node-change asynchronous alerts had been reported.
Synchro	When the synchro field is SR_Synchro, a node-change async with event type EV_NodeDead is reported for each known node. When the synchro field is SR_NoSynchro, no node-change asynchronous alerts are sent as a result of this transaction.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_NotInTable

FN_REGY_NodeChangeToRegistry

This transaction tells the registry service that the status of a node has changed. This is an internal transaction within the adapter.

Minor_function 16

Parameter_DDR

This is a pointer to the following data:

Byte	**3**	**2**	**1**	**0**
0	Node			
4	Event			
8	Reserved = 0			

Node	This identifies the IPN node whose status has changed.
Event	This identifies the event, which can be:

EV_NodeDead	Node has stopped working
EV_Rebooted	Node has completed its IPL

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_REGY_NodeChangeFromRegistry

This async transaction is passed on to all the modules that have connected for node-change asynchronous alerts.

To ensure that deadlock does not occur in the registry service, the receiver of this transaction should complete this transaction before issuing another transaction to the registry service.

Minor_function 17

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Node			
4	Event			
8	Reserved = 0			Synchro

Node	This identifies the IPN node whose status has changed.
Event	This identifies the event, which can be: **EV_NodeDead** Node has stopped working **EV_Rebooted** Node has completed its IPL
Synchro	The synchro field is SR_Synchro if the transaction is sent as a result of a FN_REGY_ConnectForNodeChange or FN_REGY_DiscForNodeChange transaction in which the synchro field was SR_Synchro. Otherwise, the synchro field is SR_NoSynchro.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_REGY_ConnectForErrorLogging

This transaction tells the registry service the node and service number of the error logger.

Minor_function 18

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Node			
4	Service			

Node	This identifies the IPN node to which error logs should be sent.
Service	This identifies the service of the IPN node to which error logs should be sent.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_TableFull

FN_REGY_DiscForErrorLogging

This transaction tells the registry service that the error logger is no longer interested in receiving error logging records.

Minor_function 19

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Node			
4	Service			

Node	This identifies the IPN node to which error logs had previously been sent.
Service	This identifies the service of the IPN node to which error logs had previously been sent.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_NotInTable

FN_REGY_LogErrorToRegistry

This transaction requests the registry service to send an error logging record to the error logger.

Minor_function 20

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0 through n	Error Data			

Error Data See "FN_REGY_LogErrorFromRegistry" for the definition of error data.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_REGY_LogErrorFromRegistry

This transaction requests the error logger to log the error data supplied.

To ensure that deadlock does not occur in the registry service, the receiver of this transaction should complete this transaction before issuing another transaction to the registry service.

Minor_function 21

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Sense Format	Template	Type
4 through 16	Serial Number			
	Reserved = 0			
20	Port 1 SSA loop A	Port 2 SSA loop A	Port 1 SSA loop B	Port 2 SSA loop B
24	Count			
28 through n	Sense Data			

Type This defines the type of the sender of the error data:

TY_Disk	Disk

TY_Adapter Adapter

Template This defines the error template that should be used for logging the error data.

Sense Format This defines the format of the sense-data filed when the type is TY_Adapter, as follows:

		Byte	
0	SD_Code:	28–30	Error code
1	SD_CodeAsn:	28–30	Error code
		31	Reserved = 0
		32–46	Resource serial number
2	SD_CodeAsnCsn:	28–30	Error code
		31	Reserved = o
		32–46	Array serial number
		48–62	Component serial number
3	CfgComplete	28–30	Error code = 000000h
		31	Reserved = 0
		32	Network
			0 NI_NetworkA
			1 NI_NetworkB
		33	Loop
			0 LP_Unknown
			1 LP_Loop
			2 LP_String
		34	Legal
			0 LG_Unknown
			1 LG_Legal
			2 LG_Illegal
		35	Master
			0 MN_Unknown
			1 MN_Master
			2 MN_NonMaster
		36–39	Node Count (number of nodes including this one on this SSA network or 0xFFFFFFF if unknown)
		40–43	Initiator Count (number of initiators not including this one on this SSA network or 0xFFFFFFFF if unknown)
4		28–30	Error code
		31	Reserved = 0
		32–46	Resource serial number
		48–62	Logical Block Address

Serial Number This 15-byte ASCII character field contains the serial number of the sender.

When the type field is TY_Adapter, the format of the serial number is the ASCII card serial number (as reported in the VPD data) in bytes 4 through 11 and ASCII blanks in bytes 12 through 18.

When the type field is TY_Disk, the format of the serial number is as defined in "FN_ISALMgr_Inquiry" on page 161.

Port n This is the SSA address of the node in error on this port of the adapter card, or FFh if the disk in error is not connected to this port. If the type is TY_Adapter, this field is FFh.

Count This is the number of sense data bytes that follow this field.

Sense Data If the type is TY_Disk, this is the SCSI sense data from the disk.

Note: The sense data received from the SSA-SCSI attachment to the disk is in big-endian format and this is returned in the parameter_DDR data without any byte swapping.

If the type is TY_Adapter, this is adapter status data. This includes the adapter error code in bytes 30 through 28 (byte 28 is the most significant byte). The remainder of the sense data may include additional information; the contents are defined by the Format field.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_REGY_ConnectForResrcChange

This transaction informs the registry service that the client is interested in resource-change asynchronous alerts for resources of the specified owning module type (OMT).

Minor_function 22

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Node			
4	Service			
8	Reserved = 0		Synchro	Owning Module Type

Node This identifies the IPN node to which resource-change asynchronous alerts should be sent.

Service This identifies the service of the IPN node to which resource-change asynchronous alerts should be sent.

Owning Module Type
This identifies the type of resource for which resource-change asynchronous alerts should be sent.

Synchro When the synchro field is SR_Synchro, the registry service sends, before this transaction completes, an FN_REGY_ResrcChangeFromRegistry transaction for all resources of the specified owning module type currently registered.

When the synchro field is SR_NoSynchro, only resource state changes registered after this transaction has completed are reported by a FN_REGY_ResrcChangeFromRegistry transaction.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_TableFull

FN_REGY_DiscForResrcChange

This transaction informs the registry service that the client is no longer interested in resource-change asynchronous alerts for resources of the specified owning module type.

Minor_function 23

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Node			
4	Service			
8	Reserved = 0		Synchro	Owning Module Type

Node This identifies the IPN node to which resource-change asynchronous alerts had previously been sent.

Service This identifies the service of the IPN node to which resource-change asynchronous alerts had previously been sent.

Owning Module Type
This identifies the type of resource for which resource-change asynchronous alerts had previously been sent.

Synchro When the synchro field is SR_Synchro, a FN_REGY_ResrcChangeFromRegistry transaction is sent before the completion of this transaction for each resource of the specified owning module type known by the registry service.

When the synchro field is SR_NoSynchro, no transactions are sent as a result of this transaction.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_NotInTable

FN_REGY_ResrcChangeToRegistry

This transaction informs the registry service about a resource change.

Minor_function 24

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Undefined			
4	Service			
8	ResourceID			
12	Reserved = 0			Change Code

Service This identifies the service of the resource change.

ResouceID This identifies the resource that has changed.

Change Code The resource can be in one of the following states:

Unknown: It is not possible to communicate with this resource and, if its presence had previously been known and it had been opened, the handle has been closed.

RS_Offline: The presence of the resource has been detected and a handle is still assigned, but I/O operations to the resource are not now possible. An example is a RAID-5 array with two or more members missing. Another situation is when a resource is removed or deconfigured from an adapter and handles open for this resource have not yet been closed.

When in this state, the only valid transactions that can be sent to this handle are:

FN_ISAL_Close

FN_ISAL_Inquiry

FN_ISALMgr_Characteristics

FN_ISALMgr_Statistics

A result field AE_Offline is returned to all other transactions.

RS_Online: The presence of the resource is known and is operational. It may or may not have been opened and a handle assigned. Even though it is operational it may not be fully functional, and some transactions may not be fully executed due to the degraded condition of the resource.

The change-code field identifies the reason for the resource change:

CC_Add:	The resource, which was previously unknown, is now in the RS_Offline state.
CC_SetOnline:	The resource, which was previously in the RS_Offline state, is now in the RS_Online state. Communication with this resource, which had a handle assigned, is now possible again.
CC_Add+CC_SetOnline:	The resource, which was previously unknown, is now in the RS_Online state. Communication is now possible.
CC_SetOffline:	The resource previously in the RS_Online state is now in the RS_Offline state. This is a cue to clients that they should now close any handles that they may have open for the resource. Failure to do so in a timely fashion could cause an adapter to hang and may keep the resource stuck in the RS_Offline state even when the resource no longer exists and really should be RS_NotKnown. I/O operations using the stale handle on the resource are failed with AE_Offline.
CC_Remove:	The resource previously in the RS_Offline state, is now in the RS_NotKnown state. There are no handles open for this resource. The resource ID is removed from the registry.
CC_SetOffline+CC_Remove:	The resource previously in the RS_Online state is now in the RS_NotKnown state. There are no handles open for this resource. The resource ID is removed from the registry.

The following change code may be passed on its own or in addition to the flag combinations given above:

CC_Changed: This indicates that some aspect of the resource's characteristics has changed. Characteristics can be obtained with the FN_ISALMgrCharacteristics transaction. Any client caching information about the resource should use this notification as a cue to refresh its cache. Changes in characteristics that may have integrity issues are not communicated in this manner.

CC_Changed is not reported with adapter code below level A000.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_InvalidRID

FN_REGY_ResrcChangeFromRegistry

This transaction informs the previously-identified service of a resource change.

To ensure that deadlock does not occur in the registry service, the receiver of this transaction should complete this transaction before issuing another transaction to the registry service.

Minor_function 25

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Node			
4	Service			
8	ResourceID			
12	Reserved = 0		Synchro	Change Code

Node This identifies the IPN node of the resource change.

Service This identifies the service of the resource change.

ResourceID This identifies the resource that has changed.

Change Code This code identifies the reason for the resource change. The states of the resource are defined in “FN_REGY_ResrcChangeToRegistry” on page 143.

CC_Add: The resource, which was previously unknown, is now in the RS_Offline state.

CC_SetOnline: The resource, which was previously in the RS_Offline state, is now in the RS_Online state. Communication to this resource, which had a handle assigned, is now possible again.

CC_Add+CC_SetOnline:
The resource, which was previously unknown, is now in the RS_Online state. Communication is now possible.

CC_SetOffline - The resource, which was previously in the RS_Online state, is now in the RS_Offline state. Communication to the resource is no longer possible but the handle is still assigned.

CC_Remove: The resource, which was previously in the RS_Offline state, is now unknown. Communication to the resource is not possible and the handle has been closed.

CC_SetOffline+CC_Remove:
The resource, which was previously in the RS_Online state, is now unknown. Communication to the resource is not possible and a handle is not assigned.

CC_Changed This indicates that some aspect of the resource’s characteristics has changed. Characteristics can be obtained with the FN_ISALMgrCharacteristics transaction. Any client caching information about the resource should use this notification as a cue to refresh its cache. Changes in the characteristics that may have integrity issues are not communicated in this manner.

CC_Changed is not reported at adapter code levels below A000.

Synchro The synchro field is SR_Synchro when the transaction is sent as a result of the synchro field being SR_Synchro in an FN_REGY_ConnectForResrcChange or FN_REGY_DiscForResrcChange transaction.

If the transaction is sent as a result of the synchro field being SR_Synchro in a FN_REGY_ConnectForResrcChange transaction, the change-code field is CC_Add, if the resource is in the RS_Offline state and is a combination of CC_Add and CC_SetOnline, if the resource is in the RS_Online state

If the transaction is sent as a result of the synchro field being SR_Synchro in an FN_REGY_DiscForResrcChange transaction, the change-code field is CC_Remove, if the resource is in the RS_Offline state and is a combination of CC_Remove and CC_SetOffline, if the resource is in the RS_Online state.

The synchro field is SR_NoSynchro when the resource change transaction is not a result of the synchro field in a FN_REGY_ConnectForResrcChange or FN_REGY_DiscForResrcChange transaction being RS_Synchro.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_RetryWhenMemory

FN_REGY_ResrcList

This transaction returns a list of resource IDs that have been added to the registry service for a particular owning module type (OMT).

Minor_function 26

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Skip		Reserved = 0	Owning Module Type

Skip This defines the number of resource-list entries that should be skipped before the first resource returned in the Receive_DDR data.

Owning Module Type

This identifies the owning module type of resources that should be reported. The owning module types are:

OM_DriverPhysicalDisk
This is for a physical SSA-SCSI disk. It is used by the host to identify disks that can perform commands, such as HardwareInquiry and Open in Service Mode, that cannot be sent to a logical disk. All physical disks have one of these entries in the registry as well as having one of the following logical disk entries. Errors are logged against resource IDs of this owning module type.

OM_NotOwned This indicates that the disk is not owned by a resource manager or by a driver. This type of disk cannot be used by a driver or resource manager and is therefore a spare disk until the owning module type is changed.

OM_DriverAdapter
Resource IDs with this OMT refer to other adapter cards. This is used to implement adapter to adapter communications needed for HACMP.

OM_DriverManualDisk
This indicates a disk that has been assigned a permanent

resource ID with a configuration tool. Only personal systems require this.

OM_DriverAutomaticDisk
This is the other type of driver-owned logical disk. This indicates that the adapter, rather than an operator, has automatically assigned a number to a disk. All new disks are initialized with this value.

OM_FastWriteFilter
The fast write caching filter owns the disk.

OM_RAID0Filter
The RAID-0 filter owns the disk

OM_RAID5Filter
The RAID-5 filter owns the disk

OM_ListAll Report resource for all owning module types

Transmit_DDR Null

Receive_DDR This is a pointer to a buffer which will receive the following data:

Byte	3	2	1	0
0	ResourceID			
4	Service Number			
8	Reserved = 0			State
12	ResourceID			
16	Service Number			
20	Reserved = 0			State
24	ResourceID			
28	Service Number			
32	Reserved = 0			State
.				
.	.			
n−8	ResourceID			
n−4	Service Number			
n	Reserved = 0			State

ResourceID These are the resource IDs of the resources with the requested owning-module type. They are sorted in ascending order.

Service Number
This identifies the service for each resource ID of the requested type on this node.

State This can be:

RS_Offline: The presence of the resource has been detected and a handle is still assigned, but communication to the resource is not now possible. When in this state, the only valid transaction that can be sent to this handle are FN_ISAL_Close, FN_ISALMgrCharacteristics, and FN_ISALMgrStatistics. A result field of AE_Offline is returned to all other transactions.

RS_Online: The presence of the resource is known and is operational. It may or may not have been opened and a handle assigned. Even though it is operational it may not be fully functional, and some transactions may not be fully executed due to the degraded condition of the resource.

Status_DDR This is a pointer to a buffer that receives the following data when result is AS_Success::

Byte	3	2	1	0
0	Count			

Count The number of entries in the received data DDR. If the Receive_DDR data length constrains the number of bytes returned, Count is the number of entries that would have been returned with a data length in a Receive_DDR that did not constrain the transfer.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_REGY_GetTempResrcID

This transaction returns a temporary resource ID that can be used by a resource manager that needs to invent a resource ID name. The resulting 32-bit field has a 24-bit number unique among all resource IDs except those of type OM_DriverManualDisk.

The upper 8 bits (the owning module type) is set to zero and the caller must fill in his owning module type before the resource ID can be used.

Minor_function 27

Parameter_DDR
None

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to a buffer which will receive the following data:

Byte	**3**	**2**	**1**	**0**
0	Resource ID			

Resource ID The resulting prototype resource ID.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_REGY_ConnectForHealthCheck

This transaction sent to the local registry service by any client that needs to be informed when a health check should be performed.

Minor_function 28

Parameter_DDR
This is a pointer to the following data:

Byte	**3**	**2**	**1**	**0**
0	Node			
4	Service			

Node This identifies the IPN node.

Service This identifies the service of the IPN node that is able to perform health checks.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_TableFull

FN_REGY_DiscForHealthCheck

This transaction is sent to the local registry service by any client that no longer needs to be informed of when health checks should occur.

Minor_function 29

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Node			
4	Service			

Node	This identifies the IPN node.
Service	This identifies the service of the IPN node that is no longer able to perform health checks.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_NotInTable

FN_REGY_HealthCheckToRegistry

This transaction is sent to the registry service by a client when a health check needs to be performed.

Minor_function 30

Parameter_DDR
Null

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_REGY_HealthCheckFromRegistry

This transaction is sent by the registry service to all the local services that are registered as being able to perform health checks. It indicates these tests should occur now. The service sends error log data to the registry service which, for detected error conditions that cause a degraded operation or require a service action, forwards it to the error logger.

To ensure that deadlock does not occur in the registry service, the receiver of this transaction should complete this transaction before issuing another transaction to the registry service.

Minor_function 31

Parameter_DDR
Null

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_REGY_SerialNumberSearch

This transaction returns the resource ID and service number of the resource identified by the serial number supplied.

Minor_function 32

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0 through 12	Serial Number			
	Owning Module Type			

Serial Number This identifies the resource for which the resource ID is requested.

Owning Module Type
This identifies the type of resource, with the requested serial number, that should be reported. If this field is zero, the resourceID of the resource of any owning module type (OMT), except OM_DriverPhysicalDisk, is reported.

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to a buffer that receives the following data:

Byte	3	2	1	0
0	ResourceID			
4	Service Number			
8	Reserved = 0			State

ResourceID This is the resource ID of the resource identified by the serial number and owning module type.

Service Number
This is the service number of the manager that controls the resource.

State The current state of the resource can be one of the following (described in "FN_REGY_ResrcChangeToRegistry" on page 143):
RS_Online
RS_Offline

Result The following result fields can be returned:

AS_Success

AE_NotInTable

FN_REGY_TestResrcsReady

This transaction returns an AS_Success result when all the known resources are ready to receive transactions. This may involve a delay while, for example, the spindle motor of a disk drive is started. If all the resources are not ready within the time period defined in the parameter_DDR, the AS_Failure result field is returned. The registry service sends FN_ISALMgr_TestResrcsReady transactions to all services that are registered to inquire if all their resources are ready.

Minor_function 33

Parameter_DDR
This is a pointer to a the following data:

Byte	3	2	1	0
0	Time			

Time This defines the maximum duration in seconds before a result field must be returned.

Transmit_DDR Null

Receive_DDR Null

Status_DDR None

Result The following result fields can be returned:

AS_Success

AE_Failure

Illegal Request (range)

FN_REGY_SetClusterNumber

This transaction identifies the cluster number of the system to the registry service. An ISAL manager can obtain this from the registry using a DC_ClusterNumber directive.

The cluster number can be in the range 0 through 2048.

The adapter assumes cluster number 0 from power on until it has been set by this transaction. Cluster number 0 is not a valid cluster number for the FN_ISAL_Fence transaction. The cluster number should only be set once.

Minor_function 34

Parameter_DDR
This is a pointer to a the following data:

Byte	3	2	1	0
0	Cluster Number			

Transmit_DDR Null

Receive_DDR Null

Status_DDR None

Result The following result field can be returned:

AS_Success

FN_REGY_TestOneResrcReady

This transaction enquires of the registry the state of a resource identified by a serial number. The resource might not be in a state that permits it to be declared to the registry service. It can therefore be used to determine if any resource manager knows about this resource and if it is spinning up, exposed, or suitable to be used.

To implement this transaction, the registry sends an FN_ISALMgr+TestOneRsrcReady transaction to all known TP_ISAL services to find out if any of them know about the resource. If they all return AE_NotInTable, then the registry returns AE_NotInTable to this transaction. If any ISAL service returns another result field, then this is returned to the FN_REGY_TestOneResrcReady transaction.

Minor_function 35

Parameter_DDR
This is a pointer to a the following data:

Byte	3	2	1	0
0 through 12		Serial Number		
	Reserved = 0			

Serial Number This identifies the resource for which the resource ID is requested.

Transmit_DDR	Null
Receive_DDR	Null
Status_DDR	None
Result	The following result field can be returned:

AS_Success	(The resource is known by the registry service and by an ISAL manager and can be used. If the resource is an array, it is not exposed, but it might be degraded or rebuilding.)
AE_NotReady	(The resource is known by an ISAL manager but it is not ready for use and has not been declared to the registry service. The resource might be a disk drive that is starting. This result is only returned if the resource is expected to become usable later.)
AE_Offline	(The resource is known by an ISAL manager but cannot be used because the array is in the offline state. An array is in this state when more than one of its members is not available.)
AE_AvoidReadWrite	(The resource is known by the registry service and a resource manager and can be used. However, read and write operations to the resource should be delayed. RAID-1 filter can return this result.)
AE_AvoidWrite	(The resource is known by the registry service and an ISAL manager and can be used. However, write operations to the resource should be delayed because a write operation would cause an array to change from the exposed to the degraded state.)
AE_NotInTable	(The resource is not known by any ISAL manager.)

Illegal Request (range)

Note: (If the transaction is rejected by AE_UnknownFunction, this should be treated as AE_NotInTable.)

FN_REGY_SyncHCheckToRegy

In response to a FN_REGY_SyncHCheckToRegy transaction, the registry service issues a FN_REGY_SyncHCheckFromRegy transaction to all the connected services. If they all return AS_Success or AE_UnknownFunction, the registry service returns AS_Success. Otherwise, the registry service returns the most serious sense data it has received by means of the Status_DDR with an AE_Failure result field.

Minor_function 36

Parameter_DDR
Null

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to the following data:

Byte	3	2	1	0
0	Length			
4−n	Sense Data			

The sense DDR consists of a 4-byte length field followed by sense information of variable length. The length of the sense data is a multiple of 4 and is less than, or equal to, 36.

Result The following result field can be returned:

AS_Success

Illegal request (range)

AE_Failure

FN_REGY_SyncHCheckFromRegy

In response to a FN_REGY_HealthCheckFromRegy transaction, a service generates a FN_REGY_LogErrorToRegistry transaction shortly afterwards. However, in response to a FN_REGY_SyncHCheckFromRegy transaction, a service determines the most serious health-check complaint. The sense data that would usually be logged to the registry is returned in the Status_DDR data, and AE_Failure is returned in the result data. If there are no health-check complaints, the service returns AS_Success.

Any service that connects for health checks receives the new FN_REGY_SyncHCheckFromRegy transaction as well as the FN_REGY_HealthCheckFromRegy transaction.

Minor_function 37

Parameter_DDR
Null

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to the following data:

Byte	3	2	1	0
0	Length			
4−n	Sense Data			

The sense DDR consists of a 4-byte length field followed by sense information of variable length. The length of the sense data is a multiple of 4 and is less than, or equal to, 36.

Only adapter errors can be returned by this means. The service receiving the FN_REGY_SyncHCheckFromRegy transaction is not permitted to perform lengthy processing (for example, that involving other transactions) before completing the transaction; such delay might cause deadlock within the adapter microcode.

Result The following result field can be returned:

AS_Success

Illegal request (range)

AE_Failure

AE_UnkownFunction

IPN Storage Access Language (ISAL) Services

The Advanced SerialRAID Adapter disk service uses the IPN Storage Access Language (ISAL) to provide access to the disks in SSA subsystems. The language is similar to SCSI; however, only the functions required by clients are included.

ISAL has a single access mode that is set when the resource is opened. The ISAL transaction that opens a resource establishes a logical connection between the master and slave for that resource. This transaction is sent to the ISAL manager service which returns a handle for that manager that is used, in subsequent transactions, to access the resource just opened. All requests that are sent to the disk service are attempted. Error recovery is performed by the ISAL server and, if this fails, the sender is not required to retry the failed request. There is no contingent allegiance mode. Error logs are reported to the error logger without the sender having to request error data. If a request fails, commands that are waiting are not rejected; they are attempted in turn.

ISAL Transactions

The ISAL transactions that the disk service handles are listed in the following table.

In addressing resources, the handle number acts as a disk number (like a SCSI LUN). The transmit and receive parameters are used to point to I/O data buffers. The function parameter is sent in the minor function code field of the transaction function word, and any other parameters are sent in the parameter field of the transaction.

A physical resource is one with owning module type OM_DriverPhysicalDisk. A logical resource is one with any other owning module type.

Table 58. ISAL Transactions

Transaction	**Minor_function**	**Valid to Logical Resource**	**Valid to Physical Resource**
FN_ISALMgr_Inquiry	40	Yes	Yes
FN_ISALMgr_HardwareInquiry	41	No	Yes
FN_ISALMgr_SetOwningModuleType	42	Yes	No
FN_ISALMgr_AssignManualResrcID	43	Yes	No
FN_ISALMgr_GetPhysicalResrcIDs	44	Yes	No
FN_ISALMgr_GetPhysSvcAndRIDs (Note 5)	64	Yes	No
FN_ISALMgr_TestResrcsReady	45	Yes	Yes
FN_ISALMgr_TestOneResrcReady	63	Yes	Yes
FN_ISALMgr_VPDInquiry	46	Yes (note 4)	Yes
FN_ISALMgr_Characteristics	47	Yes	Yes
FN_ISALMgr_Statistics	48	Yes	Yes
FN_ISALMgr_FlashIndicator	49	Yes	Yes
FN_ISALMgr_NetworkInquiry (note 5)	66	Yes	No
FN_ISALMgr_Preferences (note 5)	67	Yes	No
FN_ISALMgr_LockQuery	69	Yes	Yes
FN_ISALMgr_Open	50	Yes (note 1)	Yes (note 3)
FN_ISAL_Close	51	Yes	Yes
FN_ISAL_Read	52	Yes	Yes
FN_ISAL_Write	53	Yes	Yes (Note 2)
FN_ISAL_Format	54	No	Yes (Note 2)
FN_ISAL_Progress	55	No	Yes (Note 2)
FN_ISAL_Lock	56	Yes	Yes
FN_ISAL_Unlock	57	Yes	Yes
FN_ISAL_Test	58	Yes	Yes
FN_ISAL_SCSI	59	No	Yes
FN_ISAL_Download	60	No	Yes
FN_ISAL_Fence (note 6)	62	Yes	No
FN_ISALMgr_Flush (note 5)	68	Yes	No
FN_ISAL_InitSurf (note 2)	70	No	Yes

Notes:

1. A logical resource cannot be opened in MD_Service or MD_SCSI mode.
2. Format, Progress, Write, and InitSurf transactions are not allowed to a physical resource if the corresponding logical resource for that device is also open.

3. A physical resource cannot be opened in MD_Service mode if the corresponding logical resource for that device is currently open or if the resource is in an SSA string network unless it is the last node. A physical resource cannot be opened in MD_ISAL_HA mode.
4. Array managers do not support the FN_ISALMgr_VPDInquiry.

FN_ISALMgr_Inquiry

This transaction is sent to the resource manager requesting the serial number of the specified resource.

Minor_function 40

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Resource ID			

Resource ID This identifies the resource

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to a buffer that will receive the following data when Result is AS_Success:

Byte	3	2	1	0
0 through 12	Serial Number	Serial Number	Serial Number	Serial Number
	Reserved = 0			

Serial Number This 15-byte ASCII field contains the serial number of the specified resource. It has the following format:

• **Non-RAID Disk:**

Byte	3	2	1	0
0 through 8	Product Identifier			
12	Reserved = 0	‘D’	SSA-SCSI LUN	

Note: The ASCII character ‘D’ is reported in byte 14 of the Status_DDR data if the resource is an SSA disk drive. If the SSA device is of any other type, byte 14 is the hexadecimal digit in bits 3 through 0 of byte 0 of the SSA-SCSI Inquiry data for that device, reported as an ASCII character. For example, the Character ‘5’ is reported for a CD-ROM drive.

Product Identifier This ASCII field identifies the device attached to the SSA bus. This is the 6-byte IEEE SSA unique ID translated to a 12-character ASCII string.

SSA-SCSI LUN This ASCII field identifies the SSA-SCSI logical unit number of the resource.

- **Array resource:**

Byte	3	2	1	0
0 through 12	Array Name			
	Reserved = 0	Array Letter		

Array Name This 15-ASCII-character field identifies the array.

Array Letter This ASCII character identifies the type of filter of the array resource. The letter ‘K’ is used for a RAID-5 array.

Result The following Result fields can be returned:

AS_Success

Illegal Request (range)

AE_InvalidRID

FN_ISALMgr_HardwareInquiry

This transaction is sent to the service that manages the resource to return details about the specified resource. It returns hardware specific information. Only SSA resource managers that control physical SSA devices support this transaction. The transaction is rejected with illegal-request result if the owning module type of the resource is other than OM_DriverPhysicalDisk.

Minor_function 41

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Resource ID			
4	Reserved = 0			Immed

Resource ID This identifies the resource

Immed This field controls whether the result field is returned immediately or after error recovery. (If the disk drive motor is stopped, error recovery can take over a minute.) The field can have the following values:

HI_Immediate
If the motor is stopped, AS_Success is returned immediately with status of ST_Failed and fail code of HF_MotorFail. The adapter attempts error recovery to restart the motor after the result field is returned.

HI_NotImmediate
If the motor is stopped, full error recovery is performed before the result field is returned. This could take over a minute if the motor is stopped.

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to a buffer that receives the following data when result is AS_Success or AE_ReservationConflict:

Byte	3	2	1	0
0	Port 1 SSA loop A	Port 2 SSA loop A	Port 1 SSA loop B	Port 2 SSA loop B
4	Reserved = 0		Fail Code	Status

Port n This is the SSA address of the node on this port of the adapter card. If the resource is not connected to this port then a value of FFh is returned. These fields are valid if the result field is AS_Success or AE_ReservationConflict.

Status This reports the state of the resource and is valid if the result field is AS_Success. It has the following definition:

ST_Good Good.

ST_Failed Failed. In this state, if the resource is a target on a SSA link, a Test Unit Ready SSA command is rejected with check-condition status. This could be due to a failure of POST2, a stopped motor, or any degraded mode condition.

ST_LossRedundancy
In this state, the resource has lost some redundancy, for example, loss of redundant power or cooling. The ISAL manager

determines this by sending a SSA-SCSI Inquiry command to the resource.

ST_FormatInProgress
Format operation has not yet completed.

Fail Code This provides more details if the status is ST_Failed:

HF_MotorFail	The motor is stopped
HF_Unknown	No more details of the failure are available.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_InvalidRID

AE_ReservationConflict

AE_Offline

AE_OfflineTimeout

AE_Failure

AE_RequirePhysical

FN_ISALMgr_SetOwningModuleType

This transaction is sent to the manager of the resource to set the owning module type (OMT) for the specified resource. This causes the ID for the resource to change and the new OMT to be written in the label record of the ISAL reserved area. This transaction is not used to change the OMT to OM_DriverManualDisk. FN_ISAL_AssignManualResrcID is used for that purpose.

If the resource is in the open state when this transaction is received, AS_success is returned in the result field, the new resource ID is created, and the old resource goes to the RS_Offline state. The transaction is rejected with illegal-request result if the owning module type of the resource is OM_DriverPhysicalDisk.

Minor_function 42

Parameter_DDR
The data descriptor is a pointer to the following data:

Byte	3	2	1	0
0	Old Resource ID			
4	Reserved = 0			Owning Module Type

Old Resource ID
This specifies the current resource for which the owning module type should be set.

Owning Module Type
This defines the type of disk service that controls the resource.

Transmit_DDR Null

Receive_DDR Null

Status_DDR The data descriptor is a pointer to the following data when result is AS_Success:

Byte	3	2	1	0
0	New Resource ID			

New Resource ID
This specifies the resource's new ID

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_InvalidRID

AE_MediumError

AE_HardwareError

AE_ReservationConflict

AE_FencedOut

AE_Offline

AE_TableFull

AE_FormatDegraded

AE_FormatInProgress

AE_Failure

AE_NonIsal

AE_RequireLogical

The resource manager responds to this transaction by removing the old resource ID from the registry, getting a new temporary resource ID (by using a FM_REGY_GetTempResrcID command), setting the new OMT into it, and adding this to the registry.

FN_ISALMgr_AssignManualResrcID

This transaction is sent to the service that manages the resource to change a resource ID and owning module type. The owning module type is changed to type OM_DriverManualDisk and this is written in the label record of the ISAL reserved area.

If the resource is in the open state when this transaction is received, AS_Success is returned in the result field, the new resourceID is created, and the old resource goes to the RS_Offline state. The transaction is rejected with illegal-request result if the owning module type of the resource is OM_DriverPhysicalDisk.

Minor_function 43

Parameter_DDR
The data descriptor is a pointer to the following data:

Byte	3	2	1	0
0	Old Resource ID			
4	New Resource ID			

Old Resource ID
This specifies the current resource's ID

New Resource ID
This specifies the resource's new ID. This must have an OMT of OM_DriverManualDisk in the format Ox04HHNNNN where:

Field	**Value**
HH	00
NNNN	disk number (a value of 0000 is valid but should be avoided)

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_MediumError

AE_HardwareError

AE_ReservationConflict

AE_FencedOut

AE_Offline

AE_FormatDegraded

AE_FormatInProgress

AE_Failure

AE_NonIsal

AE_RequireLogical

AE_TableFull

The resource manager responds to this transaction by removing the old resource ID from the registry and adding the new one (using the FN_REGY_ResrcChangeToRegistry transaction for both actions). If the act of adding the new resource ID results in a return of AE_InvalidRID, this means that the new resource ID is already in use and an error is reported to the user.

FN_ISALMgr_GetPhysicalResrcIDs

This transaction is used to translate a logical resource ID into its physical members. This function returns a list of resource IDs that are of type OM_DriverPhysicalDisk. The transaction is rejected with illegal-request result if the owning module type of the resource is OM_DriverPhysicalDisk.

Minor_function 44

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Logical resource ID			

Logical resource ID
This identifies the logical resource ID that is to be translated.

Transmit_DDR Null

Receive_DDR This is a pointer to a buffer that receives the following data:

Byte	3	2	1	0
0	Physical resource ID			
4	Physical resource ID			
.				
n	Physical resource ID			

Physical resource ID
This is a list of physical resource IDs that make up the logical resource ID

Status_DDR This is a pointer to a buffer that receives the following data when result is AS_Success:

Byte	3	2	1	0
0	Count			

Count The number of entries in the received data DDR.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_InvalidRID

AE_Offline

AE_RequireLogical

FN_ISALMgr_GetPhysSvcAndRIDs

This transaction is used to translate the ID of a logical resource into its physical members. This function returns a list of resource IDs that are of the type OM_DriverPhysicalDisk and the service number that owns this resource.

The transaction is rejected with Illegal Request if the owning module type of the resource is OM_DriverPhysicalDisk.

Minor_function 64

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Logical Resource ID			

Logical Resource ID

This identifies the logical resource ID that is to be translated.

Transmit_DDR Null

Receive_DDR This is a pointer to a buffer that receives the following data:

Byte	3	2	1	0
0	Service number			
4	Physical resource ID			
8	Service number			
12	Physical resource ID			
.	.			
n-4	Service number			
n	Physical resource ID			

Service number

This identifies the service that owns each resource in the list.

Physical resource ID

This is a list of physical resource IDs that make up the logical resource ID.

Status_DDR This is a pointer to a buffer that receives the following data when result is AS_Success:

Byte	3	2	1	0
0	Count			

Count The number of entries in the received data DDR.

Result The following result fields can be returned:

AS_Success	
Illegal Request (range)	
AE_InvalidRID	
AE_Offline	
AE_UnknownFunction	(not supported on the PCI SSA 4-Port RAID Adapter)
AE_RequireLogical	

FN_ISALMgr_TestResrcsReady

This transaction is used to test that all the resources that are known to and controlled by the resource manager have started and are operational.

Minor_function 45

Parameter_DDR
Null

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success	(All known resources are operational)
AE_Failure	(One or more resources controlled by this manager is not yet operational. This might be a disk drive that has not reached its operating speed.)

FN_ISALMgr_TestOneResrcReady

The registry service sends this transaction to each ISAL manager to enquire about the state of a resource identified by a serial number. The resource might not be in a state that permits it to be declared to the registry service. It can therefore be used to determine if the resource manager knows about this resource and if it is starting, exposed, or available for use.

Minor_function 63

Parameter_DDR
This is a pointer to a the following data:

Byte	3	2	1	0
0 through 12	Serial Number			
	Reserved = 0	Serial Number		

Transmit_DDR Null

Receive_DDR Null

Status_DDR None

Result The following result field can be returned:

AS_Success	(The resource is known by the registry service and by an ISAL manager and can be used. If the resource is an array, it is not exposed, but it might be degraded or rebuilding.)
AE_NotReady	(The resource is known by an ISAL manager but it is not ready for use and has not been declared to the registry service. The resource might be a disk drive that is starting. This result is only returned if the resource is expected to become usable later.)
AE_Offline	(The resource is known by an ISAL manager but cannot be used because the array is in the offline state. An array is in this state when more than one of its members is not available.)
AE_AvoidReadWrite	(The resource is known by the registry service and an ISAL manager and can be used. However, read and write operations to the resource should be delayed. RAID-1 filter can return this result.)
AE_AvoidWrite	(The resource is known by the registry service and an ISAL manager and can be used. However, write operations to the resource should be delayed because they will cause the array to move from an exposed to the degraded state.)
AE_NotInTable	(The resource is not known by any ISAL manager.)
Illegal Request (range)	

Note: If the transaction is rejected with result AE_UnknownFunction, this should be treated as AE_NotInTable.

FN_ISALMgr_VPDInquiry

This transaction is sent to the disk service to obtain Vital Product Data of the resource identified by the resource ID field.

Minor_function 46

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	ResourceID			
4	Reserved = 0		Page Code	EVPD

ResourceID This identifies the resource for this transaction.

EVPD The Enable Vital Product Data (EVPD) field controls whether the data returned is standard inquiry data or individual VPD pages. EVPD can be:

VP_NoEVPD	Standard VPD inquiry data is returned.
VP_EVPD	The VPD inquiry data of the page identified by the page-code field is returned.

Page Code This identifies the page of vital VPD inquiry data to be returned. Page 00h identifies the pages that can be returned.

Transmit_DDR Null

Receive_DDR This is a pointer to a buffer that receives the Vital Product Data. This is the same data as that returned to the SSA-SCSI Inquiry command which is defined in the functional specification of the resource.

Status_DDR This is a pointer to a buffer that receives the following data when result is AS_Success:

Byte	3	2	1	0
0	Count			

Count The number of bytes in the received data DDR.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_InvalidRID

AE_HardwareError

AE_Offline

AE_OfflineTimeout
(only if there is one or more opened MD_ISAL_HA handles on the resource)

AE_FormatDegraded

FN_ISALMgr_Characteristics

This transaction is sent to the disk service to obtain the blocksize and capacity of the resource identified by the resource ID field.

The size returned does not include the area of the disk that is reserved for use by the adapter.

Minor_function 47

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	ResourceID			

ResourceID This identifies the resource for this transaction.

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to a buffer that receives the following data when result is AS_Success:

Byte	3	2	1	0
0	Number of Blocks			
4	Bytes per Block			
8	Number of Reserved Blocks			
12	Flags			

Number of blocks	This field identifies the number of blocks available for user data.
Bytes per Block	This field identifies the blocksize of the user data.
Number of Reserved Blocks	This field identifies the number of blocks in the ISAL reserved area that are available. This does not include the blocks that the manager may be using for its own use, for example, for a label record.
Flags	This field contains bit significant values: **CF_PFA (bit 1=1)** Disk has reported a predictive failure analysis condition and it should be replaced.

CF_Download (bit 2=1)
Download of new disk microcode is or is about to take place.

CF_ExtendedRecovery (bit 3=1)
Extended error recovery is being performed on the disk.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_InvalidRID

AE_HardwareError

AE_ReservationConflict

AE_FormatInProgress

AE_FormatDegraded

FN_ISALMgr_Statistics

This transaction is sent to the disk service to obtain statistics on the transactions executed for this adapter by the resource identified by the resource ID field. The statistics are cumulative from power-on, or adapter reset, and wrap on an overflow.

Minor_function 48

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	ResourceID			

ResourceID This field identifies the resource for this transaction.

Transmit_DDR Null

Receive_DDR Null

Status_DDR This field is a pointer to a buffer that receives the following data when result is AS_Success:

Byte	3	2	1	0
0	Number of Reads			
4	Number of Writes			
8	Number of Blocks Read			
12	Number of Blocks Written			

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_InvalidRID

FN_ISALMgr_FlashIndicator

This transaction is sent to the disk service to flash a light on the resource identified by the resource ID field.

Minor_function 49

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	ResourceID			
4	Flash			

ResourceID	This field identifies the resource for this transaction.
Flash	When the flash field is 0h, the light does not flash. When the flash field is nonzero, the light flashes continuously: one second on, one second off.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_Offline

AE_InvalidRID

AE_HardwareError

AE_ReservationConflict

AE_FormatInProgress

AE_OfflineTimeout

AE_NotSupported

FN_ISALMgr_NetworkInquiry

This transaction is sent to the disk service to determine on which network the resource is attached. This is required for HA adapters that require the member disks to be attached on the same loop.

Minor_function 66

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Resource ID			

Resource ID This identifies the resource for this transaction.

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to a buffer that receives the following data when result is AS_Success:

Byte	3	2	1	0
0	Network ID			

Network ID This identifies the SSA loop or string on which the resource is attached. it can be one of the following:

NI_NetworkA
In an SSA loop or string attached to adapter SSA interface A

NI_NetworkB
In an SSA loop or string attached to adapter SSA interface B

NI_NullNetwork
No network applicable.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_UnkownFunction
(Returned by the PCI SSA 4-Port RAID Adapter)

AE_InvalidID

AE_Offline

FN_ISALMgr_Preferences

This transaction is sent to the disk service to enquire about the preferred mode of operation. This is particularly useful to the fast-write filter to optimize its destaging of data.

Minor_function 67

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Resource ID			

Resource ID This identifies the resource for this transaction.

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to a buffer that receives the following data when result is AS_Success:

Byte	3	2	1	0
0	Valid mask			
4	Word 0			
8	Word 1			
.	.			
116	Word 28			
120	Word 29			

Valid mask Each bit set in this field indicates that the corresponding word contains valid information. If the resource manager has no preferences, it can return all the bits in this field to zero. Bit 31 is reserved.

Words 0 through 29
Each word has an assigned meaning:

0 Destage Quantization

Recommended amount of data that should be destaged for optimum performance. Useful for the fast-write filter.

1 Destage offset

Recommended offset of the start of destaged data for optimum performance. Useful for the fast-write filter.

2 Queue depth

Recommended depth of queue of transactions to keep the resource busy.

3 Geometry sector

This word can be used to recommend as different value for OS/2 for the Cylinder/Head/Sector geometry.

4 Write queue depth

Recommended depth of queue of write transactions for this resource. This is used by the fast-write filter to restrict the number of queued write transactions to one; this increases the possibility of transactions being coalesced into full-stride writes. Other filters do not set the valid bit for word 4; therefore, the device driver includes both write and read transactions queued either using the value of word 2 or its own algorithm.

5–29 Not currently assigned.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_UnkownFunction
(This function might be returned by resource managers on adapters that do not support the function or by managers that do not have any preferences.)

AE_InvalidRID

FN_ISALMgr_LockQuery

This transaction is sent to the disk service to determine to which adapter or system a resource is locked.

This transaction is supported only on adapters that perform locking without issuing a Reserve transaction to the disk drive. This transaction is not supported on the PCI SSA 4-Port RAID Adapter.

Minor_function 69

Parameter_DDR
The data descriptor is a pointer to the following data:

Byte	3	2	1	0
0	Resource ID			

ResourceID This identifies the resource for this transaction.

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to a buffer that receives the following data when result is AS_Success:

Byte	3	2	1	0
0	Reserved = 0		Cluster Number	

Byte	3	2	1	0
4 through 11	AdapterID			

If the resource is not locked, the entire status data is zero. If it is locked, it is either locked to a cluster number or an adapter ID and that field is returned.

Result The following result fields can be returned:

AS_Success

AE_UnkownFunction
(Returned by the SSA 4-Port RAID Adapter)

AE_Offline

Illegal Request (range)

AE_FormatInProgress

AE_FormatDegraded

AE_NonIsal

FN_ISALMgr_Open

This transaction is sent to the disk service to request that a resource is opened. It returns a handle to be used to address the requested resource.

Minor_function 50

Parameter_DDR
The data descriptor is a pointer to the following data:

Byte	3	2	1	0
0	Resource ID			
4	Reserved = 00h	Sharing Mode	Access Type	Operation Mode

Resource ID ID number of the resource requested to be opened.

Operation Mode

MD_ISAL
IPN Storage Access Language (ISAL)

MD_ISAL_HA

The IPN Storage Access Language (ISAL) is to be used as for the MD_ISAL_HA operation mode, but whenever access to the resources is lost (which may only be temporary) a FN_REGY_ResrcChangeToRegistry transaction is sent with change code

CC_SetOffline and transactions to the resource are terminated with AE_Offline. In the HA manager, the client filter needs to be aware of any loss of a resource even if it is temporary.

In operation mode MD_ISAL, temporary access to a resource is not immediately reported as access may be restored after an interval following, for instance, an SSA network transient or network reset. The device driver should use MD_ISAL operation mode and array filters in an HA environment will use MD_ISAL_HA.

MD_ISAL_HA mode is not supported on the PCI SSA 4-Port RAID Adapter. It is also not supported for physical resource IDs.

If the SSA network is illegal when an attempt is made to open a resource in MD_ISAL_HA mode, the transaction is rejected with an AE_Failure result.

MD_SCSI
SCSI pass-through. When a resource is in SCSI pass-through mode transactions other than SCSI sent to the returned handle are rejected with illegal-request result. If this mode is requested, the Open transaction is rejected with illegal-request result if it is sent to any resource ID that is not of the owning module type OM_DriverPhysicalDisk.

MD_Service
Service Mode. If this mode is requested, the Open transaction is rejected with Illegal Request, if it is sent to any resource ID that is not of the owning-module type OM_Driver_PhysicalDisk.

Certain conditions do not allow the resource to be open in MD_Service mode:

AE_Logopen	If the associated logical resource is currently open.
AE_SSAString	If the resource is in an SSA string rather than an SSA loop, but it is not the last node in the string

AE_ReservationConflict	If the associated logical resource is currently locked. (Not supported on the PCI SSA 4-Port RAID Adapter.)
AE_FencedOut	If the associated logical resource is currently fenced out. (Not supported on the PCI SSA 4-Port RAID Adapter.)

When a resource is in service mode, the adjacent SSA ports to this node are wrapped and the check light on the selected resource is turned on.

Access Type

AT_All	Read and Write transactions allowed
AT_ReadOnly	Read only
AT_WriteOnly	Write only

Sharing mode

SM_DenyAll	Deny read and write access
SM_DenyWrite	Deny write access
SM_DenyRead	Deny read access
SM_DenyNothing	Deny nothing
SM_DenyNone	Deny nothing and do not check other handles.

If opening in SM_DenyNone, the open succeeds regardless of the existence or otherwise of other handles.

If opening in any other sharing mode, the success of the open depends on the access type requested. If permission for the access type requested is denied by a sharing mode of an existing handle, the open is failed with AE_AccessDenied.

Transmit_DDR	Null
Receive_DDR	Null

Status_DDR The data descriptor is a pointer to the following data when result is AS_Success:

Byte	3	2	1	0
0	Handle			

Handle This is the number that the client should use to address the resource.

Result The following result fields can be returned:

AS_Success

AE_AccessDenied

Illegal Request (range)

AE_InvalidRID

AE_SSAString

AE_LogOpen

AE_Offline

AE_InServiceMode

AE_ReservationConflict
(only if opening in MD_Service mode)

AE_FencedOut (only if opening in MD_Service mode)

AE_OtherAdapterInServiceMode

FN_ISAL_Close

This transaction is sent to the disk service to close the resource identified by the handle field. If any transactions are active for the resource with this handle, the resource is not closed and the transaction terminates with an illegal-request result field.

If the resource being closed was in service mode, it is returned to normal mode before the close is completed. This may involve unwrapping SSA links of adjacent nodes.

If the resource being closed was locked before the ISAL_Close transaction, it remains locked at the end of the transaction.

If the resource can be closed and it is not open to other clients, any data or metadata that is held for any of its members are flushed to those members so that they are synchronized. That is, the service behaves as if it had received a FN_ISAL_Flush transaction immediately before the FN_ISAL_Close transaction. Any errors encountered during this flushing are ignored and the result field returned to the FN_ISAL_Close transaction is that which would have been returned if the flushing had not been attempted.

Minor_function 51

Parameter_DDR

The data descriptor is a pointer to the following data:

Byte	3	2	1	0
0	Handle			

The handle field identifies the resource for this transaction.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_ISAL_Read

This transaction is sent to the disk service to read the specified blocks from the resource identified by the handle field.

Minor_function 52

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Handle			
4	Address			
8	Count			
12	Extended Flags	Reserved = 0	Flags	Priority

Handle This identifies the resource that is to be read

Address This is the logical block address of the data to be read

Count This is the number of logical blocks to be read

Priority Reserved

Flags This field has bit-significant values. Multiple flags can be set. Bit values not specified are ignored.

FF_Verify Verify data. No data is transferred to the client. The manager validates that the data could be read if requested. For an array, data is read from the members, and might be reconstructed from the other members, but it is not transferred to the client.

This option is available primarily for service functions to physical disks or to verify that data in an array can be read. If it is issued to a fast-write resource, an attempt might be made to destage cached data first. It is not recommended that FF_Verify is used for fast-write resources because this might incur write operations to the disk which might not have been intended.

FF_ExtendedFlags
The extended-flags field is not zero.

FF_Split Data is allowed to be received out of order.

FF_ReadDisk Data must be read from the device and not from a cache.

| This is ignored by the RAID-5
| filter.

FF_ISALReservedArea

This flag causes the data to be read from the area of the disk drive reserved for ISAL. This is a separately addressed area of the resource starting at address zero. It follows the access type and sharing modes defined when the resource is opened.

The blocks that can be read are from address zero to the number of reserved blocks reported in FN_ISALMgr_Characteristics, minus one. The client may use these blocks as needed. The label record and fence sector are not visible through this interface.

Extended Flags
This field has bit-significant values. Multiple flags may be set. Bit values of the flag field not specified are ignored.

The flag field must have the FF_ExtendedFlags bit = 1b if any of the extended flags are:

EF_NoRetryOnError
When this flag is not set and the transaction reports AE_Medium error, all blocks up to the address in status have been sent to the requestor. Setting EF_NoRetryOnError avoids excessive retry time after a disk medium error but does not force all blocks up to the failing block to be sent to the requestor.

EF_Override Read operations are allowed to the resource even when it is reserved to another host or fenced out from this host.

EF_NoDestage Data is not to be destaged for fast write before the requested data is read.

Transmit_DDR Null

Receive_DDR This is a pointer to the buffer that receives the read data. The length of this buffer must be equal to or greater than the total number of bytes in the logical blocks requested if data is to be transferred by the transaction.

Status_DDR This is a pointer to the following data that is returned if the result field is AE_Warning or AE_MediumError:

Byte	3	2	1	0
0	Reserved = 0			Hint Flags
4	Address			

Hint Flags

RF_ReassignWarn
This flag, when set, indicates that the logical block identified by the address field should be reassigned. The logical block address must be within the range of the blocks requested in the Read transaction. All blocks up to this address must have been sent to the client.

Address This is the address of the logical block that should be reassigned when the Application_result field indicates either AE_Medium Error or when it contains AS_Warning and the hint-flag field is RF_ReassignWarn or RF_RewriteWarn.

Result The following result fields can be returned:

AS_Success

AE_ReservationConflict

AS_Warning

AE_HardwareError

AE_NotReady

AE_MediumError

AE_AccessDenied

AE_InvalidSignature

Illegal Request (range)

AE_Offline

AE_FencedOut

AE_FormatInProgress

AE_FormatDegraded

AE_OfflineTimeout

AE_BadBlockLRC

FN_ISAL_Write

This transaction is sent to the disk service to write the specified blocks to the resource identified by the handle field. The transaction is rejected with illegal-request result if the owning module type of the resource is OM_DriverPhysicalDisk and the corresponding logical resource is currently open.

Minor_function 53

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Handle			
4	Address			
8	Count			
12	Extended Flags	Reserved = 0	Flags	Priority

Handle	The identifies the resource that is to be written
Address	This is the logical block address where the data is to be written
Count	This is the number of logical blocks to be written
Priority	Reserved

Flags
This field has bit-significant values. Multiple flags can be set. Bit values not specified are ignored.

FF_Verify	Verify that, after writing the data, it can be read back (with reconstruction if necessary, in the case of an array).
FF_ExtendedFlags	The extended-flags field is not zero.
FF_FastWrite	The transaction can be completed before the data is written to the disk.
FF_Split	Data is allowed to be written on the disk out of order. For RAID-5 arrays, a resource-dependent-value attribute can be set, from the configurator, to allow splits on an aligned 4K page even when the FF_Split bit is off. (see “RAID-5 Filter” on page 98.)
FF_ISALReservedArea	The data is written to the area of the disk reserved for ISAL. This is a separately addressed area of the resource starting at address zero. It follows the access type and sharing modes defined in the open of the resource. The blocks that can be written are from address zero to the number of reserved blocks reported in FN_ISALMgr_Characteristics, minus one. The client can use these blocks as needed. The label record and fence sector are not visible through this interface.

Extended Flags
This field has bit significant values; multiple flags may be set. Bit values of the flag field not specified are ignored.

The flag field must have the FF_ExtendedFlags bit = 1b if any of the Extended Flags are set.

EF_Override
Allow a write to the resource even when locked to another host or fenced out from this host.

Transmit_DDR This is a pointer to the transmit buffer. The length of this buffer must be equal to or greater than the total number of bytes of the logical blocks requested.

Receive_DDR Null

Status_DDR This is a pointer to the following data, which is returned when the result field is AE_Warning or AE_MediumError:

Byte	3	2	1	0
0	Reserved = 0			Hint Flags
4	Address			

Hint Flags

RF_ReassignWarn
This flag, when set, indicates that the logical block identified by the address field should be reassigned. The logical block address must be within the range of the blocks specified in the Write transaction.

Address This is the address of the logical block that should be reassigned when the result field indicates either AE_Medium Error or when it contains AS_Warning and the hint flag field is RF_ReassignWarn or RF_RewriteWarn.

Result The following result fields can be returned:

AS_Success

AE_ReservationConflict

AS_Warning

AE_HardwareError

AE_NotReady

AE_MediumError

AE_AccessDenied

AE_InvalidSignature

Illegal Request (range)

AE_Offline

AE_FencedOut

AE_WriteProtect

AE_LogOpen

AE_FormatInProgress

AE_FormatDegraded

AE_BadBlockkLRC

AE_BadSequenceNumber

AE_OfflineTimeout
(only if there is more than one open MD_ISAL_HA handle on the resource)

FN_ISAL_Format

This transaction is sent to the disk service to start formatting the entire disk in the resource identified by the handle field. AS_Success is returned if formatting can be started. The FN_ISAL_Progress transaction can be used to track the progress and completion of the format operation. The transaction is rejected with illegal-request result if the owning module type of the resource is other than OM_DriverPhysicalDisk or if the corresponding logical resource ID for this device is currently open.

Minor_function 54

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Handle			
4	Blocksize			
8	Flag			

Handle	This identifies the resource for this transaction
Blocksize	This is the number of bytes in each logical block. This must be a value that is supported by the disk drive.
Flag	This field is reserved.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

AE_ReservationConflict

AE_HardwareError

AE_NotReady
Illegal Request (range)
AE_Offline
AE_FencedOut
AE_LogOpen
AE_FormatInProgress
AE_RequirePhysical

The progress of the format operation can be obtained by issuing a Progress transaction. If the format operation is aborted or cannot be completed (for example, if the disk drive is powered off before the operation completes) the disk drive enters degraded mode. A Format transaction must then be reissued and completed before the disk drive will allow reads and writes.

FN_ISAL_Progress

This transaction is sent to the disk service to determine the progress of a format operation to the resource identified by the handle field. The transaction is rejected with IllegalRequest result if the owning module type of the resource is other than OM_DriverPhysicalDisk.

Minor_function 55

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Handle			

Handle This identifies the resource for this transaction

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to the following data when result is AS_Success:

Byte	3	2	1	0
0	Percent			

The percent field contains the percentage of a format operation that has been completed, as an unsigned integer from 0 to 99. A value of −1 is returned if a format operation is not in progress.

Result The following result fields can be returned:

AS_Success
Illegal Request (range)
AE_Offline
AE_HardwareError

AE_LogOpen

AE_RequirePhysical

FN_ISAL_Lock

This transaction is sent to the filter to reserve exclusively to this client the resource identified by the handle field.

When a resource is locked, the lock is maintained for the following conditions:

- A lock comes into effect when an FN_ISAL_Lock is received for a resource that is not currently locked and a path exists from the client (for example, the device driver) to the resource.
- A lock is removed when:
  - An FN_ISAL_Unlock is executed successfully from the locking client and the path to the resource exists
  - An FN_ISAL_Unlock with the UL_Forced flag on is executed successfully from any client and the path to the resource exists
  - The client either unlocks or closes the resource and the path to the resource is lost. If the resource is held open and locked, the lock is not removed when the path to the resource is lost, but the lock might not be effective.
    - The firmware need not attempt to retain the lock if a resource is closed with a lock in place and the path intact and the path is subsequently lost.
    - A lock is not effective if the path from the locking client to the resource is broken (regardless of what the client does following loss of the path). Another client can access the resource or place its own lock.
    - If a path is reestablished and the lock has not been lost according to the rules above, it is the responsibility of the software immediately above the break in the path to reestablish the lock (assuming that the lock has not been broken by another client).
- The disk is formatted.
- The lock owning adapter is reset.
- The owning module type of the resource is changed.

A 'path is lost', that is, communication to the resource is not possible, if the microcode attempts, and issues, a CC_Remove change code. If the resource is held open, the microcode cannot issue a CC_Remove, but the path is still lost. A transient CC_Offline that is followed by a CC_SetOnline (caused, for example, by an SSA network reset) is not a loss of path.

The Advanced SerialRAID Adapter supports locking to either an AdapterID or to a cluster number. If locking is requested to a cluster number, the only adapters permitted to share that cluster number are adapters within the same system. All adapters sharing the same cluster number together constitute the 'locking agent' for the above rules governing when locks are maintained and cluster number based locking is used.

The following transactions are not permitted on the Advanced SerialRAID Adapter if they are addressed to a resource of type OM_DriverPhysicalDisk:

- FN_ISAL_Lock
- FN_ISAL_Unlock
- FN_ISAL_Fence
- FN_ISAL_Flush

When the associated logical resource is locked to another initiator, the following transactions are executed on a Advanced SerialRAID Adapter when addressed to a resource of type OM_DriverPhysicalDisk:

- FN_ISALMgr_HardwareInquiry
- FN_ISALMgr_Characteristics
- FN_ISALMgr_FlashIndicator
- FN_ISAL_Progress

The resource identified by the handle field must be a disk drive, an array, or a Fast-write resource; otherwise the transaction is rejected with an AE_NonIsal result field.

Minor_function 56

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Handle			
4	Reserved = 0			Type

Handle This identifies the resource for this transaction

Type This defines the type of lock to be performed:

LT_Normal
Lock resource to the cluster number of the client, if available, or to the adapter ID if it is not.

LT_AdapID
Lock resource to the adapter ID of the client.

Note: On the Advanced SerialRAID Adapter, an FN_ISAL_Lock transaction with a Parameter_DDR of only 4 bytes causes the resource to be locked as though LT_Normal was requested.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

AE_ReservationConflict

Illegal Request (range)

AE_Offline

AE_FencedOut

AE_HardwareError

AE_FormatInProgress

AE_FormatDegraded

AE_WriteProtect

AE_NonIsal

AE_RequireLogical

AE_OfflineTimeout
(only if there is one or more open MD_ISAL_HA handles on the resource)

FN_ISAL_Unlock

This transaction is sent to the disk service to terminate the previous reservation to this client of the resource identified by the handle field.

This transaction is rejected with AE_IllegalRequest if it is addressed to a resource of type OM_DriverPhysicalDisk. This transaction has no effect on the list of systems fenced out for the resource even when the flag is UL_Forced.

Minor_function 57

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Handle			
4	Reserved = 0			Flag

Handle This identifies the resource for this transaction.

Flag The following flags control whether the unlock should be unconditional or not:

UL_Normal The unlock operation is unsuccessful if the resource is already locked to another client.

UL_Forced The resource is unlocked even if it is locked to another client. This can be implemented by resetting the resource.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

AE_ReservationConflict

Illegal Request (range)

AE_Offline

AE_FencedOut

AE_HardwareError

AE_FormatInProgress

AE_FormatDegraded

AE_RequireLogical

AE_OfflineTimeout
(only if there is one or more open MD_ISAL_HA handles on the resource)

FN_ISAL_Test

This transaction is sent to the disk service to test the ability of the resource identified by the handle field to execute transactions. This might involve internal tests being performed by the resource.

Minor_function 58

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Handle			
4	Reserved = 0			Diagnostic

Handle This identifies the resource for this transaction.

Diagnostic

TT_Test No internal test is performed in the resource

TT_Diag Internal tests are performed in the resource

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

AE_NotReady

AE_ReservationConflict

AE_HardwareError

Illegal Request (range)

AE_Offline

AE_FencedOut

AE_FormatDegraded

AE_FormatInProgress

AE_OfflineTimeout
(only if there is one or more open MD_ISAL_HA handles on the resource)

FN_ISAL_Download

This transaction is sent to the disk service to download code to the resource. If the resource ID is not for a resource of owning module type OM_DriverPhysicalDisk, the transaction is rejected with an illegal-request result.

Execution of transactions sent to this physical disk after this transaction are delayed until after the Download transaction has completed.

Minor_function 60

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Handle			
4	Count			
8	Reserved = 0			Flag

Handle	This identifies the resource for this transaction
Count	This is the number of bytes of the download
Flag	This controls if the downloaded code is saved in nonvolatile storage:

DL_Save Downloaded code is saved in nonvolatile storage

DL_NoSave	Downloaded code is not saved and will be lost when power is removed from the resource.

Transmit_DDR	This is a pointer to the transmit buffer
Receive_DDR	Null
Status_DDR	Null
Result	The following result fields can be returned:

AS_Success
AE_NotReady
AE_ReservationConflict
AE_HardwareError
Illegal Request (range)
AE_Offline
AE_FencedOut
AE_FormatInProgress
AE_RequirePhysical

FN_ISAL_Fence

This transaction removes or adds hosts to the list of those fenced for the resource identified by the handle field. The list of hosts fenced for that resource at the end of the transaction is returned.

Fencing provides a means of preventing access by one or more hosts that are suspected of malfunctioning or should be excluded from access to the resource for other reasons. In a two-initiator network, one processor can exclude the other by using the Lock transaction. With more than two initiators, the Lock transaction cannot be used for this purpose, because it excludes all hosts but one.

When an initiator is fenced out for the resource, the following transactions are rejected with result field AE_FencedOut:

- All transactions that require the resource to be opened before execution, except FN_ISAL_Fence with FF_Force or FN_ISAL_Close.
- ISAL Manager transactions FN_ISALMgr_SetOwningModuleType, and FN_ISALMgr_AssignManualResourceID.

If the host attempts to fence itself out from the resource, the transaction is failed with an illegal-request result field.

The transaction is rejected with IllegalRequest if it is addressed to a resource of type OM_Driver_PhysicalDisk.

Hosts are identified for fencing by their cluster number which is set in the adapter by a FN_REGY_SetClusterNumber transaction. The cluster number can be in the range 1 through 2048. The adapter defaults to cluster number 0 from when power is turned on to it until it receives the FN_REGY_SetClusterNumber transaction. The maximum number of cluster numbers that can be fenced in or fenced out is 96.

Unless the resource identified by the handle field is a disk drive, an array, or a fast-write resource, it is rejected by an AE_NonIsal result field.

Minor_function 62

Parameter_DDR
This is a pointer to the following data:

Byte	**3**	**2**	**1**	**0**
0	Handle			
4	Reserved = 0	Force	Count	

Handle This identifies the resource for this transaction.

Force

FF_Normal If the resource is fenced out from this initiator, the transaction is not executed and is terminated with AE_FencedOut result field.

FF_Force The transaction is executed even when the resource is fenced out from this initiator. FF_Force can be used to forcibly change the list of initiators fenced out that have been set by an initiator that has failed. It will also cause any reservation for that resource to be released.

Count This is the number of bytes of data that are pointed to by the Transmit_DDR parameter. If the count field is zero, the list of initiators fenced for the resource is returned without any change.

Transmit_DDR This is a pointer to the following data:

Byte	3	2	1	0
0	Mask Count		Modifier	ListFormat
4	Reserved = 0	Change	Cluster Number (1)	
8	Reserved = 0	Change	Cluster Number (2)	
.				
4n	Reserved = 0	Change	Cluster Number (n)	
4n+4	Mask Cluster Number (1)		Reserved = 0	
4n+8	Mask Cluster Number (3)		Mask Cluster Number (2)	
.				
4n+4+2m	Mask Cluster Number (m)		Mask Cluster Number (m-1)	

ListFormat This defines whether the list of systems is to be interpreted as systems fenced out or systems fenced in from the resource. If the list-format parameter of the current list of fenced systems is not in the same format as required by this transaction, the list format is changed to the new format and the previous list is deleted.

- **FL_FenceOut** The system identified by the following cluster-number field is to be added or removed to the list of initiators fenced out for this resource.

- **FL_FenceIn** The system identified by the following cluster-number field is to be added or removed from the list of fenced initiators **not** fenced out for this resource.

Modifier

- **FM_Change** The systems identified by the cluster numbers supplied are to be added or removed from the list of fenced clusters.

- **FM_CompareAndSwap**
  A mask of cluster numbers is provided in the Transmit_DDR data. Clusters are only removed from or added to the list of fenced clusters when the list of mask cluster numbers matches the list of fenced cluster numbers at the start of the transaction. The

cluster numbers in the list of mask cluster numbers must be in ascending order.

Mask Count This is a count of the number of bytes of Transmit_DDR data used for mask cluster numbers. This includes a 2–byte reserved field.

Change This controls if the cluster number is added or removed from the list:

FC_Add The system identified by the following cluster number field is added to the list of fenced initiators for this resource.

FC_Remove The system identified by the following cluster number field is removed from the list of fenced initiators for this resource.

Cluster Number
The cluster number identifies the system that is to be added or removed from the list of those fenced out. The cluster number can be in the range 1 through 2048.

Receive_DDR This is a pointer to the following data:

Byte	3	2	1	0
0	Cluster Number (1)		ListFormat	
4	Cluster Number (3)		Cluster Number (2)	
8	Cluster Number (5)		Cluster Number (4)	
.				
2n-2	Cluster Number (n)		Cluster Number (n-1)	

Status_DDR This is a pointer to a buffer that receives the following data:

Byte	3	2	1	0
0	Count			

Count This identifies the number of bytes in the Receive_DDR area.

Result The following result fields can be returned:

AS_Success

AE_ReservationConflict

AE_FencedOut

AE_Offline

AE_ClusterNumberNotKnown

AE_HardwareError

AE_NotReady

Illegal Request (range)

AE_FormatDegraded

AE_FormatInProgress

AE_OfflineTimeout
(only if there is one or more opened MD_ISAL_HA handles on the resource)

AE_NonIsal

AE_RequireLogical

AE_UnknownFunction

If a resource is fenced out and also reserved to another initiator, transactions to that resource are rejected with AE_ReservationConflict result field.

FN_ISAL_SCSI

This transaction is sent to the service that manages the resource to issue a raw SCSI command to the resource identified by the handle field. Unlike all other transactions, the only recovery performed by the adapter is for SSA link errors.

The resource must have been opened in SCSI pass-through mode for this transaction to be executed. If the resource ID is not for a resource of owning module type OM_DriverPhysicalDisk, the transaction is rejected with an illegal-request result.

SSA-SCSI linked commands should not be used. The link bit in the command descriptor block must be 0.

The following restrictions apply to disk devices only. The CDB length must be 6, 10, or 12 bytes. SCSI Reserve and Release commands are not accepted. Any change to the mode pages may be over-ridden by other adapter microcode after an indeterminate period.

On the Advanced SerialRAID Adapter, sense data is fetched from the device and held in the adapter when a command is terminated with check condition status. This sense data is returned to the host if the next command is Request Sense or returned in the Status_DDR data if the SI_AutoSense flag is set. The sense data is retained in the adapter only until completion of the next FN_ISAL_SCSI transaction.

Minor_function 59

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Handle			
4	Flags		Reserved = 0	Identifier
8–n	Command Descriptor Block			

Handle This identifies the resource for this transaction. The physical resource identified can contain several logical units (LUNs).

Identifier This field identifies the SSA_SCSI logical unit number to which the resource manager should send the SCSI command. The format of this field is as defined for SSA_SCSI:
- If bit 7 is 1, the field identifies the target routine
- Bits 6 through 0 identify the logical-unit routine.

Flags The following flags can be set:

SI_Override When this is set the transaction is executed even when the resource is fenced out from this adapter or locked to another adapter.

SI_NonIdem When this is set the transaction is terminated and AE_NonIdem result is returned if the execution of the command is interrupted by a link reconfiguration.

SI_AutoSense When this is set, sense data is returned in the Status_DDR if the command is terminated with Check Condition status. If the flag is not set, the Status_DDR contains only the SCSI status field.

Command Descriptor Block
This is as defined for SCSI and can be 6, 10, or 12 bytes.

Transmit_DDR This is null or a pointer to any data or parameters to be sent to the device.

Receive_DDR This is null or a pointer to a buffer for any data received from the device.

The Transmit_DDR and Receive_DDR fields cannot both be non-zero. These are used by the resource manager to determine the direction of data transfer.

Status_DDR This is a pointer to a buffer that receives the following data when the result field is AE_SCSIError:

Byte	3	2	1	0
0	Count		Reserved = 0	SCSI Status
4–n	SCSI Sense Data			

SCSI Status This is the status byte as defined in SSA_SCSI that is returned by the resource. It is always non-zero. If zero status (good) is returned in the SCSI_status SSA message, the result field is AS_Success and no data is sent to the buffer pointed to by the Status_DDR field.

Count Number of bytes that follow this field

SCSI Sense data
Sense data from the resource after a command is terminated in check condition SCSI status.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_SCSIError

AE_Offline

AE_FencedOut

AE_NonIdem

AE_ReservationConflict

AE_NotSupported

AE_RequirePhysical

FN_ISAL_Flush

This transaction requests the service to flush any data or metadata for the resource so that the members of the resource are synchronized. This allows the adapter to be changed or adapters and disks to be powered off without the risks of having to rebuild arrays or losing data because of the removal of power. When the members are synchronized, any NVRAM contents are not critical.

The transaction causes the service to write any data and metadata required to its members to get them synchronized. The service then issues an FN_ISAL_Flush to each of its members. Services that do not have anything to flush, for example, the base disk service, return AS_Success. When all the members have returned AS_Success to each FN_ISAL_Flush, AS_Success is returned to this transaction.

After the array or filter is synchronized, it might remain synchronized for an indeterminate period, for example, until the next FN_ISAL_Write is received, or while an FN_ISAL_Write is not yet complete.

On an Advanced SerialRAID Adapter, this transaction is rejected with AE_IllegalRequest if it is addressed to a resource of type OM_DriverPhysicalDisk.

Minor_function 68

Parameter_DDR
This is a pointer to a the following data:

Byte	3	2	1	0
0	Handle			

Handle This identifies the resource for this transaction.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

AE_Failure (Flush of resource failed)

AE_FlushComponentFailure
(Flush of member failed)

AE_Offline

Illegal Request (range)

AE_OfflineTimeout
(only if resource opened in MD_ISAL_HA)

AE_UnknownFunction

AE_RequireLogical

FN_ISAL_InitSurf

This transaction allows the user to easily initialize large areas of the disk surface to a known pattern of all zeroes. If the target disk is block LRC protected, the data written will have the correct block LRC generated. Zero data is written to the disk by a series of write operations and, unlike FN_ISAL_Format, there is no low level disk format executed. The FF_ISALReservedArea is not changed by this transaction.

This transaction is intended primarily for use after formatting a disk in 'Block LRC protected' mode prior to using the disk as a member of a RAID array or for any situation where data could be read before being written. Otherwise, RAID-5 would encounter a block LRC error on blocks read during an initial RAID-5 rebuild.

This transaction can only be addressed to a physical ISAL resource ID, and the corresponding logical resource ID cannot be open at the same time.

Minor_function 70

Parameter_DDR
This is a pointer to a the following data:

Byte	3	2	1	0
0	Handle			
4	Logical Block Address			
8	Count			

Handle This identifies the resource for this transaction.

Logical block Address
This identifies the starting logical block address

Count The number of blocks to be written. This should be less than 128K to avoid possible timeout problems due to excessive execution time of the command to the disk.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

AS_Warning

AE_LogOpen

AE_FencedOut

AE_ReservationConflict

AE_FormatInProgress

AE_FormatDegraded

AE_InvalidRID

AE_Offline

AE_HardwareError
(Note that this includes medium errors and any data transfer error)

AE_UnknownFunction

AE_NotSupported

Illegal request (range)

AE_RequirePhysical

Adapter Service

The adapter service can be used for adapter-only transactions and for transactions to be issued to other adapters. The service number is fixed for the adapter service. The adapter service supports the following transactions:

Table 59. Adapter Transactions

Transaction	Minor_function
FN_ADAP_Control	79
FN_ADAP_TransferFromHost	80
FN_ADAP_TargetTransfer	83
FN_ADAP_TransferToHost	81
FN_ADAP_ConnectForHostTransfer	84
FN_ADAP_DisconnectForHostTransfer	85
FN_ADAP_GetClusterNumber	86
FN_ADAP_AdapterHealthCheck	90
FN_ADAP_ListSSANodes	91
FN_ADAP_QueryNodes	93
FN_ADAP_GetAdapterUID	94
FN_ADAP_SetTime	95
FN_ADAP_SetMasterPriority	96
FN_ADAP_GetMasterPriority	97
FN_ADAP_GetSupportLevel	98
FN_ADAP_QueryPort	99
FN_ADAP_ForceWrap	100
FN_ADAP_GetStatistics	101

FN_ADAP_TransferFromHost

This transaction is sent to the adapter service to request that data is sent to one or more systems. When the cluster-number field is FFFFh, the data is sent to the adapter service on all other nodes connected using FN_ADAP_TargetTransfer transactions. When the cluster number is any other value, a single FN_ADAP_TargetTransfer transaction is sent to the adapter service on that cluster.

If the type is TT_VSC, the service returns successful completion to the FN_ADAP_TransferFromHost transaction when it has received a completion (successful or unsuccessful) to all the FN_ADAP_TargetTransfer transactions it has sent or timed out waiting for a completion. If the completion is unsuccessful, the device driver of the other system will also have been monitoring its adapter and will have detected an error. If the type is TT_DataTransfer and the Cluster Number is not FFFFh, the application result either indicates successful completion, or why no FN_ADAP_TargetTransfer transaction could be sent or the application result returned by that transaction. If the cluster number is FFFFh for a TT_DataTransfer transaction type, the service returns successful completion to the FN_ADAP_TransferFromHost transaction when it has received a completion (successful or unsuccessful) to all the FN_ADAP_TargetTransfer transactions it has sent or timed out waiting for a completion.

Minor_function 80

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Cluster Number		Reserved = 0	Type
4	Count			
8–n	Parameter Data			

Cluster Number
This identifies the cluster to which data should be sent. If the value is FFFFh, data is sent to all clusters attached. The cluster number must be FFFFh if the type is TT_VSC.

If the cluster number specified has multiple adapters, the adapter service chooses to which adapter it sends the FN_ADAP_TargetTransfer transaction.

Type This can be the following:

TT_VSC Volume-status-change (VSC) type requires that the data is transmitted to all attached clusters. This can be used to inform other systems of a change in status of a particular resource.

TT_DataTransfer
Data is to be transmitted to the specified cluster (or all clusters if the cluster number is FFFFh). This can be used to send data between systems (target mode).

TT_CNUM This indicates that the data transmitted consists of the cluster number. This is transmitted to all other clusters when the cluster-number field is FFFFh.

Count This defines the number of bytes of data to be sent. The maximum value is 512. The location of this data is pointed to by the transmit_DDR. The count field must be an even number.

Parameter Data
If the type is TT_VSC, parameter data is a 16–byte

field that includes the 15–byte ASCII serial number of the resource to which the broadcast data refers. If the type is TT_DataTransfer, this field contains miscellaneous fields including a definition of the originating cluster.

Transmit_DDR This is a pointer to the data in the buffer that is to be transmitted. The maximum Data_length is 512 bytes.

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_MissingCluster

AE_RoutingError

AE_RemoteTimeout

AE_TargetNot Available

AE_TargetReceiverFull

AE_TargetTransferTooLarge

AE_Failure

FN_ADAP_TargetTransfer

This transaction is sent to the adapter service on the node identified in the cluster-number field. If the cluster number is FFFFh, the transaction is sent to the adapter service on all nodes that are attached. The data transmitted is that identified in the earlier FN_ADAP_TransferFromHost transaction. The receiving service sends an FN_ADAP_TransferToHost transaction to the host service identified by a previous FN_ADAP_ConnectForHostTransfer transaction.

If the type is TT_VSC, the receiving service returns a successful-completion result to the FN_ADAP_TargetTransfer transaction when it has received a completion to the FN_ADAP_TransferToHost transactions it has sent (successful or unsuccessful) or if no host node has connected for host transfers. If this receiving service is not able to complete the FN_ADAP_TransferToHost transaction successfully, the adapter continually presents an error status of SS_VSC. The host device driver must reset the adapter card to recover from this situation.

If the type is TT_DataTransfer, the result returned indicates the success or otherwise of executing the transaction.

Minor_function 83

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Cluster Number		Reserved = 0	Type
4	Count			
8–n	Parameter Data			

Cluster Number
This identifies the cluster to which data should be sent. If the value is FFFFh, data is sent to all clusters attached. The cluster number must be FFFFh if the type is TT_VSC.

If the cluster number specified has multiple adapters, the adapter service chooses to which adapter it sends the FN_ADAP_TargetTransfer transaction.

Type This can be the following:

TT_VSC Volume-status-change (VSC) type requires that the data is transmitted to all attached clusters

TT_DataTransfer
Data is to be transmitted to the specified cluster (or all clusters if the cluster number is FFFFh).

TT_CNUM This indicates that the data transmitted consists of the cluster number. This is transmitted to all other clusters when the cluster-number field is FFFFh.

Count This defines the number of bytes of data to be sent. The maximum value is 512. The location of this data is pointed to by the transmit_DDR. The count field must be an even number.

Parameter Data
If the type is TT_VSC, parameter data is a 16–byte field that includes the 15–byte ASCII serial number of the resource to which the broadcast data refers. If the type is TT_DataTransfer, this field contains miscellaneous fields including a definition of the originating cluster.

Transmit_DDR This is a pointer to the data that is to be transmitted. The maximum Data_length is 512 bytes.

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_Failure

AE_TargetNot Available

AE_TargetReceiverFull

AE_TargetTransferTooLarge

FN_ADAP_TransferToHost

This transaction is sent to the host service identified by a previous FN_ADAP_ConnectForHostTransfer transaction. The data transmitted is that identified in the earlier FN_ADAP_TargetTransfer transaction. If the type field is TT_VSC and, after a timeout period, no completion is received from the host service, good-completion result is returned to the FN_ADAP_TargetTransfer transaction and the adapter continually presents an error status of SS_VSC. The host device driver must reset the adapter card to recover from this situation.

Minor_function 81

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Cluster Number		Reserved = 0	Type
4	Count			
8–n	Parameter Data			

Cluster Number

This identifies the cluster to which data should be sent. If the value is FFFFh, data is sent to all clusters attached. The cluster number must be FFFFh if the type is TT_VSC.

If the cluster number specified has multiple adapters, the adapter service chooses to which adapter it sends the FN_ADAP_TargetTransfer transaction.

Type This can be the following:

TT_VSC Volume-status-change (VSC) type requires that the data is transmitted to all attached clusters

TT_DataTransfer

Data is to be transmitted to the specified cluster (or all clusters if the cluster number is FFFFh).

TT_CNUM This indicates that the data transmitted consists of the cluster number. This is transmitted to all other clusters when the cluster-number field is FFFFh.

Count This defines the number of bytes of data to be sent. The maximum value is 512. The location of this data is pointed to by the transmit_DDR. The count field must be an even number.

Parameter Data
If the type is TT_VSC, parameter data is a 16–byte field that includes the 15–byte ASCII serial number of the resource to which the broadcast data refers. If the type is TT_DataTransfer, this field contains miscellaneous fields including a definition of the originating cluster.

Transmit_DDR This is a pointer to the data that is to be transmitted. the maximum data length is 512 bytes.

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_Failure

AE_TargetNot Available

AE_TargetReceiverFull

AE_TargetTransferTooLarge

FN_ADAP_ConnectForHostTransfer

This transaction informs the adapter service of the service number in the host node that is able to receive FN_ADAP_TransferToHost transactions. Only a single host node can be connected at a time for each type of target transfer data.

Minor_function 84

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Node			
4	Service			
8	Timeout		Cluster Number	
12	Reserved = 0			Type

Node This is the IPN node that can receive the data.

Service This is the service on that node that can receive the data for the specified type of target transfer.

Timeout This is the time in seconds that the adapter service should use to timeout FN_ADAP_TransferToHost transactions that are sent to the host.

Cluster Number
This is the cluster number of the node.

Type This defines the type of target transfer transactions for which the host node is connecting. It can be:
- TT_VSC
- TT_DataTransfer
- TT_CNUM

Note: If the type field is not included in the Parameter_DDR data (parameter length 12 bytes), the adapter assumes that the host node is connecting for TT_VSC type of transfers.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:
- AS_Success
- Illegal Request (range)

FN_ADAP_DiscForHostTransfer

This transaction informs the adapter service of the service number in the host node that is no longer able to receive FN_ADAP_TransferToHost transactions of the type specified.

Minor_function 85

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Node			
4	Service			
8	Reserved = 0		Cluster Number	
12	Reserved = 0			Type

Node This is the IPN node that can no longer receive the data.

Service This is the service on that node that can no longer receive the data for the specified type of target transfer.

Cluster Number
This is the cluster number of the node.

Type This defines the type of target transfer transactions for which the host node is connecting. It can be:
TT_VSC
TT_DataTransfer
TT_CNUM

Note: If the type field is not included in the Parameter_DDR data (parameter length 12 bytes), the adapter assumes that the host node is connecting for TT_VSC type of transfers.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_ADAP_GetClusterNumber

This transaction is sent from an adapter service on one node to another node to obtain the cluster number of that node.

Minor_function 86

Parameter_DDR
Null

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to a buffer that receives the following data:

Byte	3	2	1	0
12	Reserved = 0		Cluster Number	

Cluster Number
This field contains the cluster number of the node that has been set by the FN_REGY_SetClusterNumber transaction. If this has not yet been set, a cluster number of 0 is returned (0 means undefined). The cluster number can be in the range 1 through 127.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_ADAP_AdapterHealthCheck

This transaction is sent to the adapter to report any adapter errors. Only degraded type errors can be reported in the Status_DDR data.

If the adapter has knowledge of multiple degraded errors that could be reported, only the lowest adapter error code is reported.

Minor_function 90

Parameter_DDR
Null

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to a buffer that receives the following data when result is AS_Success:

Byte	3	2	1	0
0	Reserved = 0	Adapter Error Code		

Adapter Error Code
This field contains the adapter error code of 6 hexadecimal characters. These codes are the same as are reported in adapter sense data when logging an error to the error logger. Byte 0 is the most-significant byte of the error code.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_ADAP_ListSSANodes

This transaction is sent to the adapter to report details of all nodes on the SSA network specified that have been configured. If a network change is in progress, the transaction is terminated with a result field AE_InConfig and no other information is returned.

The list of nodes is ordered according to network topology. The list starts with the node nearest port 1 and ends if it is a loop with the node closest to port 2. If the SSA network is a string, the list is ordered by the list of nodes starting from port 1 to the end of that part of the string followed by the list of nodes starting from port 2 to the end of that part of the string.

The nodes reported are those configured at the time of the last valid SSA configuration. This may not be exactly consistent with the adapters known to all services, for example, adapter service for target mode, at the time of the transaction. If an illegal SSA network is detected during a SSA configuration by the adapter, the configuration held in the adapter that is reported by this transaction is not updated from the previous valid configuration.

Minor_function 91

Parameter_DDR
This is a pointer to a buffer that contains the following data:

Byte	3	2	1	0
0	Reserved = 0	NetworkId		

NetworkID This identifies the SSA interface on the adapter to which the SSA loop or string is attached. It can be one of the following:
NI_NetworkA
NI_NetworkB

Transmit_DDR Null

Receive_DDR This is a pointer to a buffer that receives the following data:

Byte	3	2	1	0
0 through 7	Adapter ID or device SSA UID			
8	SSA Path 1			
12	SSA Path 2			
16	Total Other Ports	Port on Path 1	Port on Path 2	Remote NetworkID
20 through n	Repeat for each adapter			

AdapterID or Device SSA UID
This 8-byte binary number uniquely identifies the node. If the node is an adapter (identified by a remote networkID of NI_NetworkA or NI_NetworkB), it is the adapterID. This is the same as the SSA UID of the modules that control the SSA links with the least significant bit (bit 0) = 0; bytes 6 and 7 are 00h.

If the node is a device (identified by a Remote NetworkID of NI_Device), the 8-byte number is the device SSA UID of the node; bytes 6 and 7 are 00h.

SSA Path 1 This 4-byte field is the path component of the SSA address field from port 1 to the node. As switches are not supported, byte 11 is the only byte used for the SSA address field; bytes 8 through 10 are zero.

Bit 7 of byte 11 is zero to indicate that there are no other bytes in the path component.

If this remote node is not attached to port 1 of the adapter, FFFFFFFFh is returned.

SSA Path 2 This 4-byte field is the path component of the SSA address field from port 2 to the remote node. As switches are not supported, byte 15 is the only byte used for the SSA address field; bytes 12 through 14 are zero. Bit 7 of byte 15 is zero to indicate that there are no other bytes in the path component.

If this remote node is not attached to port 2 of the adapter, FFFFFFFFh is returned.

Total Other Ports
This contains a value that is one less than the total number of ports implemented on the node. This is normally 1b as all nodes supported have two ports.

Port on Path 1 This identifies the number of the port on the remote node that is linked to port 1 of this adapter that is either:

PI_Port1
Port 1 of the SSA chip is being used

PI_Port2
Port 2 of the SSA chip is being used

If this remote node is not attached to port 1 of the adapter, FFh is returned.

Port on Path 2 This identifies the number of the port on the remote node that is linked to port 2 of this adapter that is either:

PI_Port1
Port 1 of the SSA chip is being used

PI_Port2
Port 2 of the SSA chip is being used

If this remote node is not attached to port 2 of the adapter, FFh is returned.

Remote NetworkID
This identifies the SSA chip on the remote node to which this SSA loop or string is attached when the remote node is an adapter or that the remote node contains a single SSA chip when the node is a device. It can be one of the following:

NI_NetworkA Connected to SSA chip A on node

NI_NetworkB	Connected to SSA chip B on node
NI_Device	Single SSA chip on node

Status_DDR This is a pointer to a buffer that receives the following data when result is AS_Success:

Byte	3	2	1	0
0	Number of Entries			
4	Reserved = 0		LoopFlag	Reserved = 0
8	ChangeCount			

Number of Entries
This is the number of nodes configured on the SSA loop or string; identification and port information of each of these is returned in the Receive_DDR data. It does not include the current adapter, so a value of zero is returned if the adapter is the only node on the SSA network.

LoopFlag This indicates whether the SSA network is a loop, a string, or an invalid configuration:

SC_Loop	SSA loop configuration
SC_String	SSA string configuration
SC_IllegalString	Invalid SSA configuration (either a loop or a string)

ChangeCount This indicates the number of times the configuration information has changed from power on. It can be used to determine if there have been any changes since the last transaction was used.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_InConfig

FN_ADAP_QueryNodes

This transaction can be used to determine what devices and adapters are on an SSA network. This might be useful when the adapter determines that the network is illegal and does not configure new devices or adapters. Other transactions report only details of a legal network or parts of an illegal network that existed before it became illegal.

If a network change is in progress, the transaction is terminated with a result field AE_InConfig and no other information is returned.

Minor_function 93

Parameter_DDR
This is a pointer to a buffer that contains the following data:

Byte	3	2	1	0
0	Reserved = 0	Query Adapter	Port	NetworkID
4	Path			

NetworkID This identifies the SSA interface on the adapter to which the SSA loop or string is attached. It can be one of the following:

NI_NetworkA

NI_NetworkB

Port This identifies the port on the SSA interface from which the Query_node SSA message should be sent. It can be:

PI_Port1
Port 1 of the SSA interface to be used

PI_Port2
Port 2 of the SSA interface to be used

Query Adapter When this field is 1, the status data refers to this adapter and not to the node identified by the path field. The path field is only used to identify the node when the query-adapter field is 0.

Path This 4-byte field is the path component of the SSA address field from the adapter to the device. Because there can be no switches, byte 7 is the only byte that is used for the SSA address field; bytes 4-6 are zero. Bit 7 of byte 7 is zero to indicate that there are no other bytes in the path component.

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to a buffer that receives the following data that is part of the Query_node_reply data that is returned by the target SSA node. This is only returned if result is AS_Success:

Byte	3	2	1	0
0	Valid Mask			
4	ChangeCode			
8	Version	ULP	Total Other Ports	Current Port
12	Reserved = 0			
16 through 23	Unique_ID			

Byte	3	2	1	0
24	Reserved = 0			Port mask
28	Reserved = 0			

Valid Mask This identifies which bytes of the Status field are valid. Bit 0 of byte 0 = 1b shows that byte 0 is valid; bit 7 of byte 3 = 1b shows that byte 31 is valid.

ChangeCode This indicates how many times the configuration information has changed since power-on time. ChangeCode can be used to determine whether any changes have occurred since this transaction was previously used.

Current Port This identifies the number of the port on the destination node that is now being used. Current Port can have one of the following values:

Pl_Port1
Port 1 of the SSA chip is being used

Pl_Port2
Port 2 of the SSA chip is being used

Total Other Ports
This contains a value that is one less than the total number of ports implemented. This will normally be 1b as devices supported all have two ports.

ULP The Upper Level protocol identifies the SSA upper-level protocol that the node supports. This can be:

80	SSA-IA/95SP (level supported by current disk drives)
FC	The Advanced SerialRAID Adapter

Version This identifies the version of the SSA transport layer that is supported by the adapter:

05h	SSA-1A/95PH+

Master Priority Bits 6-4 define the priority of the node for becoming the Master of the SSA web. A value of zero indicates that this node is not capable of becoming a Master (that is, it is not an adapter).

Bit 7 is used in SSA Query_node_reply to indicate that the node has no space to add an entry for the adapter that sent the Query_node. This is 0b for this transaction because the Query_node sent has the Disable Registration bit = 0; the node does not, therefore, add the sender of the Query_node to its adapter or configuration table.

Bits 0-3 are reserved = 0.

Unique_ID This 8-byte binary number uniquely identifies the node. It consists of two reserved bytes in bytes 22-23 (containing zeroes) followed by a 6-byte IEEE Universal Address.

Port Mask This field is a bit vector to indicate the port number of those ports operational on the target node. Bit 7 = 1b indicates that port 1 is operational; Bit 6 = 1 indicates that port 2 is operational. It can therefore be determined whether the other port of a 2-port node is operational and has another node attached to it. This can be done by inspecting the other port number in the Port Mask field from the port number on which the message was received as identified in the Current Port field.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_InConfig

AE_QNTimedOut

FN_ADAP_QueryPort

This transaction can be used to obtain operating characteristics and statistics about the specified port. If a network change is in progress, the transaction is terminated with a result field AE_InConfig and no other information is returned.

Minor_function 99

Parameter_DDR

This is a pointer to a buffer that contains the following data:

Byte	3	2	1	0
0	Destination Port	Query Adapter	Port	NetworkID
4	Path			
8	Reserved = 0			Control

NetworkID This identifies the SSA interface on the adapter to which the SSA loop or string is attached. It can be one of the following:

NI_NetworkA

NI_NetworkB

Port This identifies the port on the SSA interface from which the Query_node SSA message should be sent. It can be:

PI_Port1	Port 1 of the SSA interface to be used
PI_Port2	Port 2 of the SSA interface to be used

QueryAdapter When this field is 1, the status data refers to this adapter and not to the node identified by the path field. The port that data is requested for is the same as the port identified in the Port field. The path field is only used to identify the node when the QueryAdapter field is 0.

Destination Port
This identifies the port on the selected node that information is requested for. It can be:

PI_Port1	Port 1
PI_Port2	Port 2

Path This 4-byte field is the path component of the SSA address field from the adapter to the device. Because there can be no switches, byte 7 is the only byte that is used for the SSA address field; bytes 4-6 are zero. Bit 7 of byte 7 is zero to indicate that there are no other bytes in the path component.

Control This field controls the counters to be reset. The field has bit significant values allowing multiple resets to be allowed:

QP_CLE (bit 0)	Clear Link ERP counter
QP_CLC (bit 1)	Clear Frame counters

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to a buffer that receives the following data when result is AS_Success:

Byte	3	2	1	0
0	Unused			
4	Length			
8	Alarm Threshold		Link ERP Error Count	
12	Reserved = 0	Modes	B_quota	A_quota
16	Current speed		Supported speed	
20	Source Frame counter			

Byte	3	2	1	0
24	Cut-thru Frame counter			
28	Receive Frame Counter			

Length The number of bytes following in the sense data. if the addresses node only supports SSA_IA/95PH, only 4 data bytes follow. If the node supports SSAIA/95PH+, 24 data bytes follow.

Link ERP Error Count
This 2 byte field is the count of the number of times the Link ERP has been invoked for the selected ports since the last total or absolute reset or the last Query_port SMS with the CLE bit set. FFFFh is returned if the count is greater than or equal to 65,535.

Alarm threshold
This 2 byte field is the current setting of the Alarm Threshold for LINK ERP of the selected port.

A_Quota This 1 byte field contains an unsigned binary integer that specifies the static frame quota that a node can originate before it is satisfied.

B_Quota This 1 byte field contains an unsigned binary integer that specifies the maximum static frame quota that a node can originate for each rotation of the SAT token.

Modes This 1 byte field reports various operating conditions:

bits 2:0	Cable	This field indicates the type of cable that is attached to a port as follows:
	000b	Information is not available. This is returned if the port is not connected, if a 20 MB/s copper cable is attached, or if a 20 MB/s optical extender is attached. It is also returned if the node is not directly attached to a cable, for example disk drives.
	010b	External 40 MB/s copper cable
	011b	External 40 MB/s optical cable
bit 3	Reserved = 0	
bits 5:4	Mode	The port modes reported are:
	00b	Reserved
	01b	Wrap mode
	10b	Normal mode
	11b	Privileged mode
bit 6	Reflect	When set the node reflects SAT characters as SAT’ and SAT’ characters as SAT.

bit 7	EDUC	When set the port forwards User Defined Characters

Supported speed
A bit mask to indicate which link speed the port supports:

bit 0 Set to 1b if the port supports 20 MB/s

bit 1 Set to 1b if the port supports 40 MB/s

Current speed A bit mask to indicate the current operating speed:

bit 0 Set to 1b if the port is operating at 20 MB/s

bit 1 Set to 1b if the port is operating at 40 MB/s

All other bits are reserved.

If the port is not operational, the current speed is 0000h.

Source Frame Counter
This is an unsigned binary integer giving the number of frames that the node has originated through this port since the frame counters were last reset.

Cut-thru Frame Counter
This is an unsigned binary integer giving the number of cut-through frames that the node has transmitted through this port since the frame counters were last reset.

Receive Frame Counter
This is an unsigned binary integer giving the number of valid frames that the node has received through this port since the frame counters were last reset. The count indicates cut-thru traffic as well as frames that the node is the destination for.

The frame counters wrap to 0 when incrementing from 232 — 1. (The frame counters will wrap every 4 hours at the maximum frame rate so they need to be polled more frequently than this.)

The frame counters are reset when the parent node powers on, when the parent node receives a total or absolute reset frame or when a Query_port message with CFC = 1b is addressed to the port.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_UnknownFunction

AE_InConfig

FN_ADAP_GetAdapterUID

This transaction returns the SSA UID of the specified SSA chip on the adapter. The SSA UID of both SSA chips on the adapter are identical except for the least significant bit (bit 0).

Minor_function 94

Parameter_DDR

This is a pointer to a buffer that contains the following data:

Byte	3	2	1	0
0	Reserved = 0			NetworkID

NetworkID This identifies the SSA chip on the adapter for which the SSA UID is requested. It can be:

NI_NetworkA

NI_NetworkB

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to a buffer that will receive the following data:

Byte	3	2	1	0
0 4	SSA UID			

SSA UID This 8-byte binary number uniquely identifies the adapter using this SIC. Bytes 6 and 7 are 00h.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_ADAP_SetTime

This transaction is used to set a time value that is held in the adapter. It is used in the adapter to identify the value of the adapter internal timer at that time. The time value cannot be read by the host, but can be part of data that is saved during a dump to identify the times of the traces.

Minor_function 95

Parameter_DDR

This is a pointer to a buffer that contains the following data:

Byte	3	2	1	0
0	TimeInSeconds			

TimeInSeconds This is the time, in seconds, since EPOCH.

TimeInMilliseconds
This is the time, in milliseconds, within the second that was identified in the TimeInSeconds field.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_ADAP_SetMasterPriority

This transaction is used to set the SSA adapter master priority. It is used during SSA configuration to determine which adapter is to become the SSA master. The SSA master is the adapter that has the highest UID for the adapter with the highest master priority.

Minor_function 96

Parameter_DDR
This is a pointer to a buffer that contains the following data:

Byte	3	2	1	0
0	Reserved = 0		Master Priority	NetworkID

NetworkID This identifies the SSA interface on the adapter whose Master Priority is to be set. It can be one of the following:

NI_NetworkA

NI_NetworkB

Master Priority This is used to set the Master Priority of the SSA interface specified. It can have values 0 through 7 decimal. The default value of Master Priority is 4. It is recommended that the Master Priority be set to a value of 5 for an adapter that is required to be the SSA master.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_ADAP_GetMasterPriority

This transaction is used to fetch the SSA master priority. It is used during SSA configuration to determine which adapter is to become the SSA master. The SSA master is the adapter that has the highest UID for the adapter with the highest master priority

Minor_function 97

Parameter_DDR

This is a pointer to a buffer that contains the following data:

Byte	3	2	1	0
0	Reserved = 0			NetworkId

NetworkID This identifies the SSA interface on the adapter whose Master Priority is to be fetched. It can be one of the following:

NI_NetworkA

NI_NetworkB

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to a buffer that receives the following data when result is AS_Success:

Byte	3	2	1	0
0	Reserved = 0			Master Priority

Master Priority This is used to get the Master Priority of the SSA interface specified. It can have values 0 through 7 decimal. The default value of Master Priority is 4.

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_ADAP_GetSupportLevel

This transaction is sent from an adapter service on one node to obtain the support level of that adapter. This can be used in conjunction with the LL field in the VPD data and the device ID field to precisely identify the card and its microcode function.

Minor_function

Parameter_DDR

Null

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to a buffer which will receive the following data when result is AS_Success:

Byte	3	2	1	0
0	Reserved = 0			Support Level

Support Level This field contains the support level of the adapter and identifies the firmware on the card. The support level is:

03 Advanced SerialRAID Adapter, code level <5000

04 Advanced SerialRAID Adapter, code level 5000 – 9999

05 Advanced SerialRAID Adapter, code level ≥A000

Result The following result fields can be returned:

AS_Success

FN_ADAP_ForceWrap

This transaction can be used to wrap a specified SSA port.

Minor_function 93

Parameter_DDR

This is a pointer to a buffer that contains the following data:

Byte	3	2	1	0
0	Reserved = 0		Port	NetworkID
4	Path			

NetworkID This identifies the SSA chip on the adapter that the SSA loop or string is attached to. It can be one of the following:

NI_NetworkA

NI_NetworkB

Port This identifies the port on the SSA chip that the Query_node SSA message should be sent from. It can be:

PI_Port1
Port 1 of the SSA chip to be used

PI_Port2
Port 2 of the SSA chip to be used

Path This 4-byte field is the path component of the SSA address field from the adapter to the device. As there can be no switches, byte 7 is the only byte used for the SSA address field; bytes 4 – 6 are zero. Bit 7 of byte 7 is zero to indicate that there are no other bytes in the path component.

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_ADAP_Control

This transaction can be used to control the reporting of SSA open link errors.

Minor_function 79

Parameter_DDR
This is a pointer to a buffer that contains the following data:

Byte	3	2	1	0
0	Flags			

Flags The only control flag currently defined is:

CT_LogLoopOpen
When set, all SSA open links are reported as error logs. Reporting continues until any of the following:

- 255 open link errors reported
- 32 health checks have occurred
- CT_LogLoopOpen is reset

Transmit_DDR Null

Receive_DDR Null

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

AE_UnknownFunction (not supported below adapter code level 5000).

FN_ADAP_GetStatistics

This transaction returns adapter statistics. The only statistic currently defined is the power on hours of the fast write cache battery.

Minor_function 101

Parameter_DDR
Null

Transmit_DDR Null

Receive_DDR Null

Status_DDR This is a pointer to a buffer that receives the following data when Result is AS_Success:

Byte	3	2	1	0
0	Battery Power On Hours			
4	Battery Life			
8	Status			

Battery Power On Hours
The number of hours thatthe fast write cache battery has been powered on.

Battery Life
The number of hours that the fast write cache battery can be powered on before it is replaced.

Status Bit 0 = 1, battery is active.
Bits 1 – 31 reserved.

Result The following result fields can be returned:

AS_Success

Array-Configuration Service

The array-configuration service uses the IPN array configuration language (IACL) to define the configuration of array filters to be used in the adapter. In these transactions, Parameter_DDR and Status_DDR are used, but Transmit_DDR and Receive_DDR are not.

The array-configuration service handles the following transactions:

Table 60. Array-Configuration Transactions

Transaction	Minor_function
FN_IACL_Register	102
FN_IACL_Unregister	103
FN_IACL_Command	101

FN_IACL_Register

This transaction is issued by a filter service to declare to the array-configuration service that the filter exists. This must be sent before any configuration transactions can be issued to the array-configuration service.

Minor Function 102

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Service			
4	Reserved = 0			Filter Type

Service	This is the service number of the registering filter
Filter Type	This is the filter type of the registering filter

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_IACL_Unregister

This transaction is issued by a filter service to declare to the array-configuration service that no more transactions should be sent to this filter.

Minor Function 103

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0			Filter Type

Filter Type	This is the filter type of the registered filter

Status_DDR Null

Result The following result fields can be returned:

AS_Success

Illegal Request (range)

FN_IACL_Command

In this transaction, the real function is defined in the first word of the parameter DDR. The functions are defined on pages 231 through 285.

Minor Function 101

Parameter_DDR
This is a pointer to data that has the following format:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through n	Command Parameters			

Function This specifies the one of the following functions:

Code	**Function**
1	FC_IACLVersion
2	FC_ResrcCount
3	FC_ResrcList
4	FC_ResrcView
5	FC_CandidateCount
6	FC_CandidateList
7	FC_ResrcCreate
8	FC_ResrcDelete
9	FC_ResrcRename
10	FC_ComponentView
11	FC_ComponentExchange
12	FC_QueryMetaResrcParams
13	FC_ModifyResrcParams
14	FC_FlashIndicator
15	FC_VPDInquiry
16	FC_VPDHardwareInquiry
17	FC_CompExchCandCount
18	FC_CompExchCandList
19	FC_AdapterVPD
20	FC_SyncHealth
21	FC_Wrap
22	FC_Unwrap
23	FC_UnwrapAll
24	FC_Test
25	FC_Format

26	FC_Certify
27	FC_Read
28	FC_Write
29	FC_AdapterSN
30	FC_CacheFormat
31	FC_InitSurf
32	FC_HotSpareCfgStatus
33	FC_HotSparePoolList
34	FC_HotSparePoolView
35	FC_ReadArrayHotSpareParams
36	FC_WriteArrayHotSpareParams
37	FC_DeconfigureDisk
38	FC_CoupleArray
39	FC_UncoupleArray
40	FC_ReadUncoupledArrayMetaData
41	FC_CoupleCompCandCount
42	FC_CoupleCompCandList
43	FC_CoupleResrcCandCount
44	FC_CoupleResrcCandList
45	FC_CoupledArrayComponentView
46	FC_WriteUncoupledArrayMetaData

Filter Type This identifies the filter that is being configured, for both arrays and disks that are not in arrays. The valid filter types are:

- FT_DriverAutomaticDisk *
- FT_DriverManualDisk *
- FT_RAID0Filter
- FT_RAID5Filter
- FT_PartitioningFilter
- FT_FastWriteFilter
- FT_PhysicalDisk *
- FT_NotOwned *
- FT_HotSpare *
- FT_BlankReserved *
- FT_Disowned *
- FT_3rdCopyFilter

The filter types marked with a * are not filters but represent resources that are either unowned by or are logically attached to the system device driver. These filter types are referred to as *pseudofilters*. The other filter types are referred to as *real filters*.

FT_DriverAutomaticDisk is used by AIX, and FT_DriverManualDisk is used by PS/2 operating systems.

The filter type FT_DriverAutomaticDisk supports only the following commands:

- FC_IACLVersion
- FC_ResrcCount
- FC_ResrcList
- FC_ResrcView
- FC_ResrcCreate
- FC_ResrcDelete
- FC_CandidateCount
- FC_CandidateList

Status_DDR All FN_IACL_Command transactions return Status_DDR data. The format of this data is:

Byte	3	2	1	0
0	Unused			
4	Length			
8 through n	Command Result			

The length field is the byte count of Status_DDR data that follows this field.

FC_IACLVersion

This function returns the version number of the IACL language. This allows the array-configuration service to validate that the IACL level supported by the adapter card (array-configuration service and the RAID Filters) is correct. It also allows the array-configuration service to determine which filter types are present on the adapter. The array-configuration service returns AE_NotInTable for filters not present.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	

Function This is the function code, 1, for FC_IACLVersion

Filter Type This is the filter type to which the function is directed.

Status_DDR This is a pointer to the following data when result is AS_Success:

Byte	3	2	1	0
0	Unused			
4	Length = 4			
8	Version			

Length This is 4, showing that 4 bytes of data follow this field

Version This is a 32-bit unsigned integer that identifies the code level of the filter.

Result The following result fields can be returned:

AS_Success

AE_Failure

AE_NotInTable

Illegal Request (range)

FC_ResrcCount

This function returns the number of resources that a particular filter has created by earlier FC_ResrcCreate functions.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	

Function This is the function code, 2, for FC_ResrcCount

Filter Type This is the filter type to which the function is directed

Status_DDR This is a pointer to the following data when result is AS_Success:

Byte	3	2	1	0
0	Unused			
4	Length = 4			
8	Resource Count			

Length This is 4, showing that 4 bytes of data follow this field

Resource Count
This is a 32 bit unsigned integer that identifies the number of resources created for this filter.

Result The following result fields can be returned:

AS_Success

AE_Failure

Illegal Request (range)

FC_ResrcList

This function requests a list of resources for the specified filter and their status. The selection of resource names (serial numbers) that are required is identified in the parameter data.

| When a FT_3rdCopyFilter array is coupled with an FT_RAID1Filter or FT_RAID10Filter
| array, the FT_3rdCopyFilter is not listed on the FC_ResrcList. Status of a coupled array
| is reported with the status of the array that it is coupled to.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4	First Resource Number (n)			
8	Requested Count (m)			

Function This is the function code, 3, for FC_RescrList

Filter Type This is the filter type to which the function is directed

First Resource Number (n)
This is the ordinal number of the first resource (starting with zero) that is reported in the Status_DDR data.

Requested Count (m)
This is the number of resources from the first resource number that are to be reported.

Status_DDR This is a pointer to the following data when result is AS_Success:

Byte	3	2	1	0
0	Unused			
4	Length			
8 through 20	Reserved = 0	Serial Number (n)		
24	Couple percent	Couple status	Percent	Status
28 through 40	Reserved = 0	Serial Number (n+1)		
44	Couple percent	Couple status	Percent	Status
.				
20m-12 through 20m	Reserved = 0	Serial Number (n+m)		
20m+4	Couple percent	Couple status	Percent	Status

Length The identifies the number of bytes that follow this field (320 maximum)

Serial Number This 15-character ASCII string is the name of the array.

Status This can be one of the following:

FS_ResrcOffline
If this is a pseudofilter, this status indicates the resource is in the RS_Offline state defined on "FN_REGY_ResrcChangeToRegistry" on page 143. If it is a real filter, this status indicates that the array does not have enough members to function or it contains inconsistent members.

FS_ResrcOnline
This is only returned for a pseudofilter. It indicates the resource is in the RS_Online state defined in "FN_REGY_ResrcChangeToRegistry" on page 143.

FS_ResrcOnlineNonDeg
The array is not degraded and is fully operational.

FS_ResrcOnlineDeg
The array is degraded.

FS_ResrcOnlineRebuild
The missing member has been returned to a degraded array which is in the process of rebuilding.

FS_ResrcOnlineExposed
A member is missing from an array and no write operations have yet been required to that member.

FS_ResrcUnknown
This is the state that an array is in until N-1 members are visible for the first time.

FS_ResrcWrapped
The physical resource is wrapped

FS_RescrFormatting
The physical resource is being formatted: the percent field reports the amount currently formatted.

FS_ResrcCertifying
The physical resource is being certified: the percent field reports the amount currently certified.

FS_ResrcIniting
The physical resource is being initialized; the percent field reports the amount currently initialized.

FS_ResrcFormatFailed
Formatting the disk has failed; the percent field reports how much of the disk was formatted before the failure.

FS_ResrcCertifyFailed
Certifying the disk has failed; the percent field reports how much of the disk was certified before the failure.

FS_ResrcInitFailed
Initializing the disk has failed; the percent field reports how much of the disk was initialized before the failure.

FS_ResrcWrongCache
The cache on the Fast Write card is not the one that holds data for this disk.

FS_ResrcInaccessibleCache
A resource that is not in synchronization

has been identified but there is no cache on the adapter or the cache is not accessible.

FS_ResrcUnrecoverableDataLoss
There has been an unrecoverable data loss to the resource due to a fast write cache failure.

FS_ResrcInUse
This is only reported for an NVRAM resource. It indicates that the defined resource is associated with a known array.

FS_ResrcDormant
This is only reported for an NVRAM resource. It indicates that the defined resource is not associated with any known array.

Percent This is an integer in the range 0 through 99 indicating the percentage completion of an operation for the following fields:

FS_ResrcRebuild
Rebuilding the array

FS_ResrcFormatting
Formatting the array

FS_ResrcCertifying
Certifying the array

FS_ResrcIniting
Initializing the array

Couple Status This field is only valid if there is an FT_3rdCopyFilter array coupled to the listed array. This field may contain any of the following:

FS_ResrcCoupleOnlineNonDeg
The FT_3rdCopyFilter coupled array is not degraded and is fully operational.

FS_ResrcCoupleOffline
The FT_3rdCopyFilter coupled array does not have enough members to function or it contains inconsistent members.

FS_ResrcCoupleOnlineDegraded
The FT_3rdCopyFilter coupled array is degraded.

FS_ResrcCoupleOnlineCopying
The FT_3rdCopyFilter coupled array is still being copied.

FS_ResrcCoupleUnknown
The FT_3rdCopyFilter coupled array stays in this state when the parent array is in the FS_ResrcUnknown state.

Couple Percent An integer in the range 0 to 100 indicating the percentage completion of copying the array contents to the FT_3rdCopyFilter coupled array. It is only valid if FS_ResrcCoupleOnlineCopying is reported in the Couple Status field.

Result The following result fields can be returned:

AS_Success

AE_Failure

Illegal Request (range)

FC_ResrcView

This function is used to examine one resource of a filter in more detail. The resource name is sent in the parameter_DDR data. Details of the resource characteristics and status are returned in the status_DDR data.

When an FT_3rdCopyFilter array is coupled with an FT_RAID1Filter or FT_RAID10Filter array, the FT_3rdCopyFilter is not listed on the FC_ResrcList. Status of a coupled array is reported with the status of the array that it is to be coupled to.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Reserved = 0	Serial Number		

Function This is the function code, 4, for FC_ResrcView

Filter Type This is the filter type to which the function is directed

Serial Number This 15-character ASCII string is the name of the resource.

Status_DDR This is a pointer to the following data when result is AS_Success:

Byte	3	2	1	0
0	Unused			
4	Length = 44			
8	Component Count			
12	Resource Size			
16 through 47	Resource Dependent Values			
48	Couple percent	Couple status	Percent	Status

Length This is the number of bytes of data, 44, that follow this field

Component Count
This is the number of components that are incorporated into the resource

Resource Size This is the number of blocks available for user data

Resource Dependent Values
These resource parameters differ for each filter type. The structure for each filter type is reported by the FC_QueryMetaResrcParams function (see "FC_QueryMetaResrcParams" on page 251). All filters report the block size in bytes 19 through 16. Zeroes are returned in fields not defined.

Status This can be one of the following:

FS_ResrcOffline If this is a pseudofilter, this status indicates the resource is in the RS_Offline state defined on "FN_REGY_ResrcChangeToRegistry" on page 143. If it is a real filter, this status indicates that the array does not have enough members to function or it contains inconsistent members.

FS_ResrcOnline This is only returned for a pseudofilter. It indicates the resource is in the RS_Online state defined on "FN_REGY_ResrcChangeToRegistry" on page 143.

FS_ResrcOnlineNonDeg The array is not degraded and is fully operational.

FS_ResrcOnlineDeg The array is degraded.

FS_ResrcOnlineRebuild The missing member has been

	returned to a degraded array which is in the process of rebuilding.
FS_ResrcOnlineExposed	A member is missing from a RAID-5 array and no writes have yet been required to that member.
FS_ResrcUnknown	This is the state that an array is in until N-1 members are visible for the first time.
FS_ResrcWrapped	The physical resource is wrapped
FS_RescrFormatting	The physical resource is being formatted: the percent field reports the amount currently formatted.
FS_ResrcCertifying	The physical resource is being certified: the percent field reports the amount currently certified.
FS_ResrcIniting	The physical resource is being initialized: the percent field reports the amount currently initialized.
FS_ResrcFormatFailed	Formatting the disk has failed; the percent field reports how much of the disk was formatted before the failure.
FS_ResrcCertifyFailed	Certifying the disk has failed; the percent field reports how much of the disk was certified before the failure.
FS_ResrcInitFailed	Initializing the disk has failed; the percent field reports how much of the disk was initialized before the failure.
FS_ResrcWrongCache	Cache on the Fast Write card is not the one that holds the data for this resource.
FS_ResrcInaccessibleCache	A resource that was not in synchronization has been identified but there is no cache on the adapter or the cache is not accessible.
FS_ResrcUnrecoverableDataLoss	There has been an unrecoverable data loss to the resource due to a fast write cache failure.

FS_ResrcInUse	This is only reported for an NVRAM resource. It indicates that the defined resource is associated with a known array.
FS_ResrcDormant	This is only reported for an NVRAM resource. It indicates that the defined resource is not associated with any known array.

Percent This is an integer in the range 0 through 99 indicating the percentage completion of an operation for the following fields:

FS_ResrcRebuild	Rebuilding the array
FS_ResrcFormatting	Formatting the disk
FS_ResrcCertifying	Certifying the disk
FS_ResrcIniting	Initializing the disk

Couple Status This field is only valid if there is an FT_3rdCopyFilter array coupled to the listed array. This field may contain any of the following:

FS_ResrcCoupleOnlineNonDeg
The FT_3rdCopyFilter coupled array is not degraded and is fully operational.

FS_ResrcCoupleOffline
The FT_3rdCopyFilter coupled array does not have enough members to function or it contains inconsistent components.

FS_ResrcCoupleOnlineDegraded
The FT_3rdCopyFilter coupled array is degraded

FS_ResrcCoupleOnlineCopying
The FT_3rdCopyFilter coupled array is still being copied.

FS_ResrcCoupleUnknown
The FT_3rdCopyFilter coupled array stays in this state until n-1 of its members are visible for the first time.

Couple Percent An integer in the range 0 to 100 indicating the percentage completion of copying the array contents to the FT_3rdCopyFilter coupled array. It is only valid if FS_ResrcCoupleOnlineCopying is reported in the Couple Status field.

Result The following result fields can be returned:

AS_Success

AE_Failure

AE_BadResrcSerialNumber

Illegal Request (range)

HardwareError

ReservationConflict

Offline

FormatInProgress

FC_CandidateCount

This function reports the total number of potential candidates that are available for use in creating an array. Only those currently unused candidates that match exactly the specified type (resource dependent values) are included in the count.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 35	Resource Dependent Values			
36	Reserved = 0			Candidate type

Function This is the function code, 5, for the FC_CandidateCount function

Filter Type This is the filter type to which the function is directed

Resource Dependent Values
This field differs for each filter type (see "FC_ResrcView" on page 237 for details)

Candidate type This defines the types of resources that can be considered as candidates for this filter. It can be:

FT_NotOwned

FT_HotSpare

FT_DriverAutomaticDisk

FT_DriverManualDisk

If the parameter length is less than 40 bytes, only FT_NotOwned resources are considered as candidates.

Status_DDR This is a pointer to the following data when result is AS_Success:

Byte	3	2	1	0
0	Unused			
4	Length = 4			
8	Candidate count			

Length This is the number of bytes of data that follow this field (4)

Candidate Count
This is the number of currently unused members, with characteristics matching the resource-dependent-value field, that could be used to create the array.

Result The following result fields can be returned:

AS_Success

AE_Failure

AE_BadParameterValues

AE_InvalidCandidateRequest

Illegal Request (range)

FC_CandidateList

This function reports the serial numbers of candidate members that are available for use in creating an array. (The total number of available members is returned by the FC_CandidateCount function.) The function specifies the ordinal number of the first member and number of candidates (maximum 16) for which data is to be reported. The length field reports the number of candidates for which data is returned.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 35	Resource Dependent Values			
36	First Candidate (n)			
40	Requested Count (m)			
44	Reserved = 0			Candidate type

Function This is the function code, 6, for FC_CandidateList

Filter Type This is the filter type to which the function is directed

Resource Dependent Values
This field differs for each filter type (see "FC_ResrcView" on page 237 for details)

First Candidate (n)
This is the ordinal number of the first candidate (starting with zero) that could be used to create the array specified.

Requested count
This is the number of candidates (maximum 16), starting with the candidate specified in the first-candidate field, for which data is requested.

Candidate type This defines the types of resources that can be considered as candidates for this filter. It can be:

FT_NotOwned

FT_HotSpare

FT_DriverAutomaticDisk

FT_DriverManualDisk

If the parameter length is less than 48 bytes, only FT_NotOwned resources are considered as candidates.

Status_DDR This is a pointer to the following data when result is AS_Success:

Byte	3	2	1	0
0	Unused			
4	Length			
8 through 20	Serial Number (n)			
	Reserved = 0			
24	Reserved = 0		Percent	Status
28 through 40	Serial Number (n+1)			
	Reserved = 0			
44	Reserved = 0		Percent	Status
.				
20m-12 through 20m	Serial Number (n+m)			
	Reserved = 0			
20m+4	Reserved = 0		Percent	Status

Length This is the number of data bytes that follow this field (320 maximum)

Serial number This 15-character ASCII string is the serial number of each member.

Status This can be one of the following:

FS_CandOnline
The member is in the RS_Online state (see “FN_REGY_ResrcChangeToRegistry” on page 143).

FS_CandOffline
The member is in the RS_Offline state (see "FN_REGY_ResrcChangeToRegistry" on page 143).

Percent This field is zero.

Result The following result fields can be returned:

AS_Success

AE_Failure

AE_BadParameterValues

AE_InvalidCandidateRequest

Illegal Request (range)

FC_ResrcCreate

This function is used to create a new resource, composed from a group of members (maximum 16). The type of all resources must be OM_NotOwned. The new resource will have the name or serial number provided in the resource-serial-number field. If the filter type is FT_DriverManualDisk or FT_DriverAutomaticDisk, the memberCount must be set to zero and there are no associated member serial numbers.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Reserved = 0	Resource Serial Number		
20 through 51	Resource Dependent Values			
52	Component Count (n)			
56 through 68	Reserved = 0	Serial Number (1)		
72 through 84	Reserved = 0	Serial Number (2)		
.				
40+16n through 52+16n	Reserved = 0	Serial Number (n)		

Function This is the function code, 7, for FC_ResrcCreate

Filter Type This is the filter type to which the function is directed

Resource Serial Number
This 15-character ASCII string is the name or serial number of the resource that is created by this function.

Resource Dependent Values
This field differs for each filter type (see “FC_ResrcView” on page 237 for details)

Component Count
This is the number of members to be used to create the new resource.

Serial number These 15-character ASCII strings are the serial numbers of the members to be used to create the new resource.

Status_DDR No Status data is required for this function, but a Status_DDR that points to the following data is returned when result is AS_Success:

Byte	**3**	**2**	**1**	**0**
0	Unused			
4	Length = 0			

Result The following result fields can be returned:

AS_Success

AE_BadResrcSerialNumber

AE_BadComponentCount

AE_BadComponentSerialNumber

AE_BadParameterValues

AE_Failure

AE_SetOMTFailed

AE_NvramError

AE_InvalidCreateRequest

Illegal Request (range)

FC_ResrcDelete

This function is used to delete an existing resource.

Parameter_DDR
This is a pointer to the following data:

Byte	**3**	**2**	**1**	**0**
0	Reserved = 0	Filter Type	Function	
4 through 16		Serial Number		
	Reserved = 0			

Function This is the function code, 8, for FC_ResrcDelete

Filter Type This is the filter type to which the function is directed

Serial Number This 15-character ASCII string is the name of the resource that is to be deleted.

Status_DDR No status data is required for this function, but a Status_DDR that points to the following data is returned:

Byte	3	2	1	0
0	Unused			
4	Length = 0			

Result The following result fields can be returned:

AS_Success

AE_BadResrcSerialNumber

AE_Failure

AE_ConfirmRequired

Illegal Request (range)

FC_ResrcRename

This function is used to rename an existing resource.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16		Old Serial Number		
	Reserved = 0			
20 through 32		New Serial Number		
	Reserved = 0			

Function This is the function code, 9, for FC_ResrcRename

Filter Type This is the filter type to which the function is directed

Old Serial Number
This 15-character ASCII string is the old name or serial number of the resource.

New Serial Number
This 15-character ASCII string is the new name or serial number of the resource.

Status_DDR No status data is required for this function, but a Status_DDR that points to the following data is returned:

Byte	3	2	1	0
0	Unused			
4	Length = 0			

Result The following result fields can be returned:

AS_Success

AE_BadOldSerialNumber

AE_BadNewSerialNumber

AE_Failure

Illegal Request (range)

FC_ComponentView

This function is used to return the serial numbers of all the components or members of a resource. The number of the members is returned by the FC_ResrcView function. The request includes the ordinal number of the first member and the number of members to be reported (maximum 16). The returned length field describes the number of members reported.

It is not valid to address the FC_memberView function to a pseudofilter.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Resource Serial Number			
	Reserved = 0			
20	First Component (n)			
24	Requested Count (m)			

Function This is the function code, 10, for FC_ComponentView

Filter Type This is the filter type to which the function is directed

Resource Serial Number
This 15-character ASCII string is the name of the resource.

First Component
This is the ordinal number of the first component (starting at zero) to be reported.

Requested Count

This is the maximum number of members that should be reported starting from the identified first member

Status_DDR This is a pointer to the following data when result is AS_Success:

Byte	3	2	1	0
0	Unused			
4	Length			
8 through 20	Reserved = 0	Serial Number (n)		
24	Reserved = 0		Percent	Status
28 through 40	Reserved = 0	Serial Number (n+1)		
44	Reserved = 0		Percent	Status
.				
20(n+m)-12 through 20(n+m)	Reserved = 0	Serial Number (n+m)		
20(n+m)+4	Reserved = 0		Percent	Status

Length This is the number of data bytes that follow this field (320 maximum)

Serial number These 15-character ASCII strings are the serial numbers of each member.

Status This can be one of the following:

FS_CompPresent

This is returned if this member of the resource is present.

FS_CompNotPresent

This is returned if this member of the resource is not present.

FS_CompNotPresentDeconf

This can be returned in a RAID-5 array for a member that is deconfigured.

FS_CompNotPresentBlank

This member is a blank slot (that is, type FT_BlankReserved)

FS_CompPresentRebuild
This is a destination of a current rebuild operation.

FS_CompPresentRebuildme
This is a destination of a future rebuild operation.

FS_CompIllegalNetwork
This is returned when the members of an array are not all on the same SSA loop. If there is a single member on a different loop from the others, FS_CompIllegalNetwork is returned only for that member. If more than one member is on a different loop, FS_CompIllegalnetwork is returned for all members of the array.

Percent This field is zero.

Result The following result fields can be returned:

AS_Success

AE_Failure

AE_BadResrcSerialNumber

AE_FiltersOnly

Illegal Request (range)

FC_ComponentExchange

This function is used to replace a member of a resource with a new member, for example, to replace a faulty disk drive in a RAID-5 array. It is acceptable not to define a new replacement if one is not available; provide a null serial number instead (this should be unique if more than one null replacement is to be undertaken). Attempting to exchange a member of a degraded array is not permitted because it would cause deletion of the array; FC_ResrcDelete should be used instead.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Resource Serial Number			
	Reserved = 0			
20 through 32	Old Component Serial Number			
	Reserved = 0			
36 through 48	New Component Serial Number			
	Reserved = 0			

Function This is the function code, 11, for FC_ComponentExchange

Filter Type Filter Type to which the function is directed

Resource Serial Number
This 15-character ASCII string is the name of the array.

Old Component Serial number
This 15-character ASCII string is the name of the old component.

New Component Serial Number
This 15-character ASCII string is the name of the new component.

Status_DDR No status data is required for this function, but a Status_DDR that points to the following data is returned when result is AS_Success:

Byte	3	2	1	0
0	Unused			
4	Length = 0			

Result The following result fields can be returned:

AS_Success

AE_BadResrcSerialNumber

AE_BadOldComponentSerialNumber

AE_BadNewComponentSerialNumber

AE_DegradedArray

AE_Failure

AE_ArrayIsBroken

AE_BadExchangeCandidate

AE_FiltersOnly

Illegal Request (range)

FC_QueryMetaResrcParams

This function returns a description of the resource parameters for the specified filter type. These are used in other functions, for example, FC_ResrcView.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4	Reserved = 0			ConfAgent

Function	This is the function code, 12, for FC_QueryMetaResrcParams
Filter Type	This is the filter type to which the function is directed
ConfAgent	This field allows the filter to return different status for different environments. It can have the following values:

CA_AIX (=0)	Used for AIX environments
CA_PC (=1)	Used for PC Server environments

If the Parameter_DDR is 4 bytes long, the status defaults to that returned for CA_AIX environments.

Status_DDR This is a pointer to the following data when result is AS_Success:

Byte	3	2	1	0
0	Unused			
4	Length			
8	Fields	Reserved = 0	Max Component	Min Component
12	Offset		Size	Type
16	MinValue			
20	MaxValue			
24	Default Value			
28	StepValue			
32	Control			
36 through n	Field definitions (bytes 12–35) for each field			

Length The number of data bytes that follow this field

Min Component
This is the minimum number of components for this filter type.

Max Component
This is the maximum number of components for this filter type.

Fields
This is the number of parameters defined in the function. Each parameter is described in the same format as bytes 12 through 35 of the status data.

Type
This is the type of parameter, which can include the following:

SDS_BLOCKSIZE
SDS_DISK_NUMBER
SDS_STRIPE_SIZE
SDS_STRETCH_SIZE
SDS_MODE_FLAGS
SDS_STRIDE_SIZE
SDS_HOT_SPARE_ENABLED
SDS_HOT_SPARE_EXACT_SIZE
SDS_REBUILD_PRIORITY
SDS_SPEC_READ
SDS_MIN_LBA
SDS_MAX_LBA
SDS_MAX_WRITE_LENGTH_CACHE
SDS_ALLOW_DELETE
SDS_MIRROR_ENABLE
SDS_BAD_PTY_STRIDE
SDS_BAD_COMPONENT_STRIDE
SDS_BAD_STRIPE
SDS_NETWORK_ID
SDS_CACHE_FSW
SDS_NO_SHUTDOWN_WHEN_IDLE
SDS_DATA_SCRUB_ENABLED
SDS_DATA_SCRUB_RATE
SDS_SIZE
SDS_CACHE_SIZE
SDS_INITIALISE
SDS_DELAY
SDS_SPLIT_RESOLUTION
SDS_RO_WHEN_EXPOSED
SDS_LAZY_PARITY_WRITE
SDS_PAGE_ALIGN_SPLIT
SDS_GEOM_SECT
SDS_READ_AHEAD_ENABLE
SDS_SKIP_WR_REBUILD
SDS_SPLIT_CONFIRM
SDS_FAST_DECONFIGURE
SDS_BLOCK_LRC
SDS_SEGMENT_SIZE
SDS_NO_INITIAL_REBUILD
SDS_HOTSPARE_SPLITS
SDS_DEFINED_HOTSPARE_POOL_ONLY

SDS_HOTSPARE_PREFERRED
SDS_BYPASS_ONE_WAY
SDS_POOL_SYNCHRONISED
SDS_HOTSPARE_POOLNUM
SDS_HOTSPARES_IN_POOL
SDS_HOTSPARES_CFG_IN_POOL
SDS_HOTSPARES_MINIMUM
SDS_HOTSPARE_ENABLED
SDS_HOTSPARE_EXACT_SIZE
| SDS_UNCOPIED_STRIPS
| SDS_COPY_VERIFY_WRITES
| SDS_COPY_RATE
| SDS_ARRAY_COUPLED
| SDS_ENABLE_EARLY_RECONSTRUCT

Size This is the size in bits of the contents of the field

Offset This is the offset of this parameter from the start of the resource parameters. It must be byte aligned even though the parameter might not contain an integer number of bytes.

MinValue This is the minimum value allowed for this parameter

MaxValue This is the maximum value allowed for this parameter

Default value This is the default value used for this parameter

StepValue This is the increment allowed for this parameter

Control This contains the following control information for the parameter:

SDSF_MB	Units are MB
SDSF_KB	Units are KB
SDSF_PERCENT	Units are %
SDSF_ON_OFF	Display On/Off rather than 0/1
SDSF_BYTES	Units are bytes
SDSF_READONLY	Cannot be changed
SDSF_UNIQUE	Entries must be different
SDSF_LBA	Units are LBAs

Result The following result fields can be returned:

AS_Success

AE_Failure

Illegal Request (range)

FC_ModifyResrcParams

This function is used to modify resource parameters of a specified array.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Serial Number			
	Reserved = 0			
20 through 51	New Resource Dependent Values			

Function This is the function code, 13, for FC_ModifyResrcParams

Filter Type This is the filter type to which the function is directed

Serial number This is the serial number of the resource.

New Resource Dependent Values
This contains new data for the resource parameter for this filter. The data differs for each filter type (see “FC_QueryMetaResrcParams” on page 251 for details).

Status_DDR No status data is required for this function, but a Status_DDR that points to the following data is returned when result is AS_Success:

Byte	3	2	1	0
0	Unused			
4	Length = 0			

Result The following result fields can be returned:

AS_Success

AE_BadParameterValues

AE_Failure

AE_NotInTable

AE_FiltersOnly

Illegal Request (range)

FC_FlashIndicator

This function is used to cause the light on all the components of the resource to flash or to stop it flashing.

Parameter_DDR

This is a pointer to the following data when result is AS_Success:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Serial Number			
	Reserved = 0			
20	Flash Interval			

Function	This is the function code, 14, for FC_FlashIndicator
Filter Type	This is the filter type to which the function is directed
Serial Number	This is the serial number of the resource.
Flash Interval	When this is zero, the light does not flash. When this is nonzero the light flashes on and off continuously. The duration of each flash is approximately one second.

Status_DDR No status data is required for this function, but a Status_DDR that points to the following data is returned:

Byte	3	2	1	0
0	Unused			
4	Length = 0			

Result The following result fields can be returned:

AS_Success

AE_BadParameterValues

AE_Failure

AE_NotInTable

Illegal Request (range)

FC_VPDInquiry

This function returns the Vital Product Data (VPD) information from the resource. It is only valid for it to be sent to a resource type FT_DriverAutomaticDisk, FT_DriverManualDisk, FT_PhysicalDisk, or FT_NotOwned. It is not valid to sent it to an array filter.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Serial Number			
	Reserved = 0	Serial Number		
20	Reserved = 0		Page Code	EVPD

Function	This is the function code, 15, for FC_VPDInquiry
Filter Type	Filter Type to which the function is directed
Serial Number	This is the serial number of the resource whose VPD is requested.
EVPD	This field, Enable Vital Product Data (EVPD), controls whether the data returned is standard inquiry data or individual VPD pages. EVPD can be:

VP_NoEVPD	Standard VPD inquiry data is returned.
VP_EVPD	The VPD inquiry data of the page identified by the page-code field is returned.

Page Code	This identifies the page of vital VPD inquiry data to be returned. Page 00h identifies the pages that can be returned.

Status_DDR This is a pointer to the following data when result is AS_Success:

Byte	3	2	1	0
0	Unused			
4	Length			
8 through n	VPD Data			

Length	This is the number of bytes that follow this field.
VPD Data	This is the same data as that returned to a SSA-SCSI Inquiry command to the resource. This data is defined in the *Technical Reference* for the resource.

Result The following result fields can be returned:

AS_Success

AE_BadParameterValues

AE_Failure

AE_NotInTable

Illegal Request (range)

FC_HardwareInquiry

This function returns hardware-specific information about the specified resource. It is only valid for resource types FT_DriverAutomaticDisk, FT_DriverManualDisk, FT_PhysicalDisk, and FT_NotOwned. It is not valid to send it to an array filter.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16		Serial Number		
	Reserved = 0			

Function This is the function code, 16, for FC_HardwareInquiry

Filter Type Filter type to which the function is directed

Serial Number This is the serial number of the disk for which information is requested.

Status_DDR This is pointer to the buffer that receives the following data when result is AS_Success:

Byte	3	2	1	0
0	Port 1 SSA loop A	Port 2 SSA loop A	Port 1 SSA loop B	Port 2 SSA loop B
4	Reserved = 0			Status

Port n This is the SSA address of the node on this port of the adapter card. If the resource is not connected to this port then a value of FFh is returned. This field is valid when the result field is AS_Success or AE_ReservationConflict.

Status This reports the state of the resource and is valid when the result field is AS_Success. It has the following definition:

ST_Good Good

ST_Failed Failed. In this state, if the resource is a target on an SSA link, a Test Unit Ready SSA command is rejected with check-condition status. This might be caused by a failure of power-on self-tests, a stopped motor, or any degraded mode condition.

ST_LossRedundancy
In this state, the resource has lost some redundancy, for example, loss of redundant

power or cooling. The disk service determines this by sending a SSA-SCSI Inquiry command to the resource.

Result The following result fields can be returned:

AS_Success

AE_NotFilters

AE_Failure

AE_NotInTable

Illegal Request (range)

FC_CompExchCandCount

This function returns the number of members that are available to be exchanged into a given array. This is very similar to the FC_CandidateCount function but it returns a count of the members suitable for member exchanging into an existing array rather than for creating a new array. This is usually a smaller set of disks. The number of members suitable for member exchanging is returned when the Exchange Type field is FT_Owned (or the parameter list is 28 bytes). If the Exchange Type field is FT_HotSpare, the number of hot spare candidates for this filter is returned.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Serial Number			
	Reserved = 0			
20	Reserved = 0			Exchange Type

Function This is the function code, 17, for FC_CompExchCount

Filter Type This is the filter type to which the function is directed

Serial Number This 15-character ASCII string identifies the array for which a member exchange is required.

Exchange Type This can be one of the following:

FT_NotOwned
The count field in the status data refers to the number of members that could be exchanged.

FT_HotSpare
The count field in the status data refers to the number of hot-spare members available to this filter.

Status_DDR No status data is required for this function, but a Status_DDR that points to the following data is returned when result is AS_Success:

Byte	3	2	1	0
0	Unused			
4	Length of following data = 4			
8	Count of exchange components			

Result The following result fields can be returned:

AS_Success

AE_BadResrcSerialNumber

Illegal Request (range)

AE_Failure

AE_FiltersOnly

AE_NotInTable

AE_ArrayIsBroken

FC_CompExchCandList

This function reports the serial numbers of all the exchange members that are available to be exchanged into a specified array. This function is very similar to FC_CandidateList except it returns candidates suitable for member exchanging into the existing array rather than for creating an array. This is usually a smaller set of disks. The function specifies the ordinal number of the first candidate and the number of candidates for which serial numbers are requested.

The members suitable for member exchanging are returned when the Exchange Type field is FT_NotOwned (or the parameter length is 28 bytes). If the Exchange Type field is FT_HotSpare, the hot spare candidates for this filter are returned.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Serial Number			
	Reserved = 0			
20	First Candidate			
24	Requested Count			
28	Reserved = 0			Exchange Type

Function This is the function code, 18, for FC_CompExchCandList

Filter Type This is the filter type to which the function is directed

Serial number This 15-character ASCII string is the name of the array for which a member exchange is required.

First Candidate
This is the ordinal number of the first member (starting with zero) to be reported.

Requested Count
This is the maximum number of members to be reported.

Exchange type
This can be one of the following:

FT_NotOwned The identification of members that can be exchanged is returned in the status data

FT_HotSpare The identification of members that can be hot spares for this filter is returned in the status data.

Status_DDR This is a pointer to the following data when result is AS_Success:

Byte	3	2	1	0
0	Unused			
4	Length			
8 through 20	Reserved = 0	Serial Number (n)		
24	Reserved = 0		Percent	Status
28 through 40	Reserved = 0	Serial Number (n+1)		
44	Reserved = 0		Percent	Status
.				
20m-12 through 20m	Reserved = 0	Serial Number (n+m)		
20m+4	Reserved = 0		Percent	Status

Length This is the number of data bytes that follow this field.

Serial number This 15-character ASCII string is the serial number of each candidate.

Status This can be one of the following:

FS_CandOnline
The member is in the RS_Online state (see "FN_REGY_ResrcChangeToRegistry" on page 143).

FS_CandOffline
The member is in the RS_Offline state (see "FN_REGY_ResrcChangeToRegistry" on page 143).

Percent This field is zero.

Result The following result fields can be returned:

AS_Success

AE_Failure

AE_BadResrcSerialNumber

AE_FiltersOnly

AE_NotInTable

Illegal Request (range)

AE_ArrayIsBroken

FC_AdapterVPD

This function returns the adapter VPD information.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	

Function This is the function code, 19, for FC_AdapterVPD

Filter Type This is the filter type to which the function is directed. This must be FT_Adapter, which is not really a filter.

Status_DDR This is a pointer to the buffer that receives the following data when result is AS_Success:

Byte	3	2	1	0
0	Reserved = 0			
4	Length			
8 through n	VPD data			

Length This is the number of data bytes that follow this in field.

VPD data This is the adapter VPD data

Result The following result fields can be returned:

AS_Success

AE_NotInTable

Illegal Request (range)

FC_SyncHealth

This function returns the most significant health-check sense data of all services attached to the registry. In response to the FC_SyncHealth transaction, the array-configuration service issues a FN_REGY_SyncHealthCheckToRegy transaction to the registry service. If no sense data is to be returned, the result field is AS_Success. The result field is AE_Failure, if sense data is returned in the status data.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	

Function This is the function code, 20, for FC_SyncHealth

Filter Type This is the filter type to which the function is directed. This must be FT_Adapter, which is not really a filter.

Status_DDR This is a pointer to the buffer that receives following data when result is AS_Success:

Byte	3	2	1	0
0	Reserved = 0			
4	Length			
8 through n	Sense data			

Length This is the number of data bytes that follow this in field.

Sense data The sense data is the most significant error log data from attached services. “FN_REGY_LogErrorFromRegistry” on page 139 defines the sense data.

Result The following result fields can be returned:

AS_Success

AE_NotInTable

Illegal Request (range)

AE_Failure

FC_Wrap

This function opens the identified physical resource in service mode. This causes the SSA ports on the adjacent nodes to be wrapped. The handle returned to the array-configuration service, when the resource is opened in service mode, is not returned to the client but held by the array-configuration service pending a future FC_Unwrap or FC_UnwrapAll or until the adapter is rebooted.

The rules for resources that can be opened in service mode causing adjacent SSA ports to be wrapped are defined in “FN_ISALMgr_Open” on page 178.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Reserved = 0	Serial Number		

Function This is the function code, 21, for FC_Wrap

Filter Type This is the filter type to which the function is directed. This must be FT_PhysicalDisk, which is not really a filter.

Serial Number This is the serial number of the resource to be wrapped.

Status_DDR No status data is required for this function, but a Status_DDR that points to the following data is returned when result is AS_Success:

Byte	3	2	1	0
0	Reserved = 0			
4	Length = 0			

Result The following result fields can be returned:

AS_Success	
AE_NotInTable	(Filter not FT_PhysicalDisk or serial number not found)
AE_PhysWrapped	(Device is currently wrapped)
AE_PhysFormatting	(Device is currently formatting)
AE_PhysCertifying	(Device is currently certifying)
AE_PhysIniting	(Device is currently being initialized)
AE_AccessDenied	
AE_Failure	
AE_InvalidRID	
AE_LogOpen	
AE_SSAString	
Illegal Request (range)	
AE_InServiceMode	

FC_Unwrap

This function closes the identified physical resource that had previously been in service mode. If the resource has not previously been opened in MD_Service mode (via the FC_Wrap IACL transaction), AE_NotOpen result field is returned.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16		Serial Number		
	Reserved = 0			

Function This is the function code, 22, for FC_Unwrap

Filter Type This is the filter type to which the function is directed. This must be FT_PhysicalDisk, which is not really a filter.

Serial Number This is the serial number of the resource to be unwrapped.

Status_DDR No status data is required for this function, but a Status_DDR that points to the following data is returned when result is AS_Success:

Byte	3	2	1	0
0	Reserved = 0			
4	Length = 0			

Result The following result fields can be returned:

AS_Success

AE_NotInTable (Filter not FT_PhysicalDisk or serial number not found)

AE_NotOpen (Resource not opened via FC_Wrap)

Illegal Request (range)

AE_Failure

FC_UnwrapAll

This function closes any physical resource that had previously been in service mode.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	

Function This is the function code, 23, for FC_UnwrapAll

Filter Type This is the filter type to which the function is directed. This must be FT_PhysicalDisk, which is not really a filter.

Status_DDR No status data is required for this function, but a Status_DDR that points to the following data is returned when result is AS_Success:

Byte	3	2	1	0
0	Reserved = 0			
4	Length = 0			

Result The following result fields can be returned:

AS_Success

AE_NotInTable	(Filter not FT_PhysicalDisk or serial number not found)

Illegal Request (range)

AE_Failure

FC_Test

This function causes internal checkouts to be executed in a physical resource. It is implemented by the array-configuration service issuing a FN_ISALMgr_Open and FN_ISAL_Test followed by an FN_ISAL_Close to the identified physical resource.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function = 24	
4 through 16		Serial Number		
	Reserved = 0			
20	Reserved = 0			Type

Function This is the function code, 24, for FC_Test

Filter Type This is the filter type to which the function is directed. This must be FT_PhysicalDisk, which is not really a filter.

Serial Number This is the serial number of the resource to be tested.

| **Type** This can be:

| **TT_Test**
| No internal checkout is performed in the
| resource.

| **TT_Diag**
| Internal checkout is performed in the
| resource.

| **Status_DDR** No status data is required for this function, but a Status_DDR that points to the following data is returned:

Byte	3	2	1	0
0	Reserved = 0			
4	Length = 0			

Result The following result fields can be returned:

AS_Success	
AE_NotInTable	(Filter not FT_PhysicalDisk or serial number not found)
AE_PhysWrapped	(Device is currently wrapped)
AE_PhysFormatting	(Device is currently formatting)
AE_PhysCertifying	(Device is currently certifying)
AE_PhysIniting	(Device is currently being initialized)
AE_AccessDenied	(Copied from FN_ISALMgr_Open)
AE_InvalidRID	(Copied from FN_ISALMgr_Open)
AE_LogOpen	(Copied from FN_ISALMgr_Open or FN_ISAL_Test)
Illegal Request (range)	
AE_ReservationConflict	(Copied from FN_ISAL_Test)
AE_HardwareError	(Copied from FN_ISAL_Test)
AE_NotReady	(Copied from FN_ISAL_Test)
AE_Offline	(Copied from FN_ISAL_Test)
AE_FencedOut	(Copied from FN_ISAL_Test)
AE_FormatDegraded	(Copied from FN_ISAL_Test)
AE_FormatinProgress	(Copied from FN_ISAL_Test)

FC_Format

This function causes formatting of the physical disk to start. AS_Success is returned if formatting does start. The array-configuration service issues a FN_ISALMgr_Open and FN_ISAL_Format to the physical disk. If formatting starts successfully, the array-configuration service constructs a record that tracks the serial number of the disk.

The array-configuration service periodically issues FN_ISAL_Progress to this disk to determine the progress of the formatting. If a FC_ResrcList or FC_ResrcView transaction is issued to a disk being formatted, the progress of the format from the last FN_ISAL_Progress transaction issued is returned in the SNS_Percent field.

When formatting completes successfully, the handle is closed and the record of the disk serial number is removed. If formatting fails, the handle is closed and a record kept for that disk so that FS_ResrcFormatFailed can be returned to a subsequent FC_ResrcList or FC_ResrcView transaction. This failure record persists until the adapter is re-booted or a wrap, format or certify is issued.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Serial Number			
	Reserved = 0	Serial Number		
20	Blocksize			

Function This is the function code, 25, for FC_Format

Filter Type This is the filter type to which the function is directed. This must be FT_PhysicalDisk, which is not really a filter.

Serial Number This is the serial number of the resource to be formatted.

Blocksize This is the number of bytes in each block. This must be a value that is supported by the disk drive.

Status_DDR No status data is required for this function, but a Status_DDR that points to the following data is returned when result is AS_Success:

Byte	3	2	1	0
0	Reserved = 0			
4	Length = 0			

Result The following result fields can be returned:

AS_Success

AE_NotInTable (Filter not FT_PhysicalDisk or serial number not found)

Illegal Request (range)

AE_PhysWrapped
(Device is currently wrapped)

AE_PhysFormatting	(Device is currently formatting)
AE_PhysCertifying	(Device is currently certifying)
AE_PhysIniting	(Device is currently being initialized)
AE_Failure	
AE_AccessDenied	(Copied from FN_ISALMgr_Open)
AE_InvalidRID	(Copied from FN_ISALMgr_Open)
AE_LogOpen	(Copied from FN_ISALMgr_Open or FN_ISAL_Format)
AE_ReservationConflict	(Copied from FN_ISAL_Format)
AE_HardwareError	(Copied from FN_ISAL_Format)
AE_NotReady	(Copied from FN_ISAL_Format)
AE_Offline	(Copied from FN_ISAL_Format)
AE_FencedOut	(Copied from FN_ISAL_Format)
AE_LogOpen	(Copied from FN_ISAL_Format)
AE_FormatinProgress	(Copied from FN_ISAL_Format)

FC_Certify

This function starts the verification of every block on the physical disk. AS_Success is returned if this verification can be started successfully.

The array-configuration service issues a FN_ISALMgr_Open to the physical disk before returning the result field to the FC_Certify transaction. If the disk opens successfully, the array-configuration service constructs a record that tracks the serial number of the disk being certified and the progress of the certification.

The array-configuration service sends FN_ISAL_Reads with FF_Verify flag on to verify that all blocks on the disk can be read. If a FC_ResrcList or FC_ResrcView transaction is issued to a disk that is being certified, the progress of the certify is returned in the SNS_Percent field.

When certifying completes successfully, the handle is closed and the record of the disk serial number is removed. If certifying fails, the handle is closed and a record kept for that disk so that FS_ResrcCertifyFailed can be returned to a subsequent FC_ResrcList or FC_ResrcView transaction. This failure record persists until the adapter is re-booted or a wrap, format, or certify is issued.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16		Serial Number		
	Reserved = 0			
20	Blocksize			

Function This is the function code, 26, for FC_Certify

Filter Type This is the filter type to which the function is directed. This must be FT_PhysicalDisk, which is not really a filter.

Serial Number This is the serial number of the resource to be certified.

Blocksize This is the number of bytes in each block. This must be a value that is supported by the disk drive.

Status_DDR No status data is required for this function, but a Status_DDR that points to the following data is returned when result is AS_Success:

Byte	3	2	1	0
0	Reserved = 0			
4	Length = 0			

Result The following result fields can be returned:

AS_Success

AE_NotInTable (Filter not FT_PhysicalDisk or serial number not found)

Illegal Request (range)

AE_PhysWrapped (Device is currently wrapped)

AE_PhysFormatting (Device is currently formatting)

AE_PhysCertifying (Device is currently certifying)

AE_PhysIniting (Device is currently being initialized)

AE_Failure (Unable to begin certifying)

AE_AccessDenied (Copied from FN_ISALMgr_Open)

AE_InvalidRID (Copied from FN_ISALMgr_Open)

AE_LogOpen (Copied from FN_ISALMgr_Open)

FC_Read

This function reads a single sector from the disk identified by the serial number field.

The array-configuration service issues an FN_ISALMgr_Open, FN_ISALMgrCharacteristics, FN_ISAL_Read, and FN_ISAL_Close to the resource before returning the result field for the FC_Read function.

Because this IACL function does not involve sending a handle to identify the resource, some applications, for example, service, might prefer to use this rather than FN_ISAL_Read when reading a sector from a disk.

The function can be addressed to a resource of any filter type, but that resource must not already be open by the device driver or by another filter.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Serial Number			
	Reserved = 0			
20	Logical Block Address			
24	Reserved = 0		Flags	Priority

Function This is the function code, 27, for FC_Read

Filter Type This is the filter type to which the function is directed.

Serial Number This is the serial number of the resource to be read.

Logical Block Address
This is the logical block address of the block to be read.

Flags This field controls the type of read to be executed. These are defined in detail in "FN_ISAL_Read" on page 182.

Priority This field is reserved for future use.

Status_DDR This is a pointer to the following data when result is AS_Success:

Byte	3	2	1	0
0	Reserved = 0			
4	Length = 0			
8 through n	Data			

Length	Number of bytes of data following in the status data. Only 512-byte blocksize is supported.
Data	The data on the LBA requested.

Result The following result fields can be returned:

AS_Success	
AE_NotInTable	(Filter not FT_PhysicalDisk or serial number not found)
Illegal Request (range)	
AE_PhysWrapped	(Device is currently wrapped)
AE_PhysFormatting	(Device is currently formatting)
AE_PhysCertifying	(Device is currently certifying)
AE_PhysIniting	(Device is currently being initialized)
AE_Failure	(Unable to begin reading)
AE_AccessDenied	(Copied from FN_ISALMgr_Open)
AE_InvalidRID	(Copied from FN_ISALMgr_Open)
AE_LogOpen	(Copied from FN_ISALMgr_Open)
AE_ReservationConflict	(Copied from FN_ISAL_Read)
AE_HardwareError	(Copied from FN_ISAL_Read)
AE_NotReady	(Copied from FN_ISAL_Read)
AE_MediumError	(Copied from FN_ISAL_Read)
AE_InvalidSignature	(Copied from FN_ISAL_Read)
AE_Offline	(Copied from FN_ISAL_Read)
AE_FencedOut	(Copied from FN_ISAL_Read)

AE_FormatDegraded
(Copied from FN_ISAL_Read)

AE_FormatinProgress
(Copied from FN_ISAL_Read)

AS_Warning (Copied from FN_ISAL_Read)

FC_Write

This function writes a single sector to the disk identified by the serial number field.

The array-configuration service issues an FN_ISALMgr_Open, FN_ISALMgrCharacteristics, FN_ISAL_Write, and FN_ISAL_Close to the resource before returning the result field for the FC_Write function.

Because this IACL function does not involve sending a handle to identify the resource, some applications, for example, service, might prefer to use this rather than FN_ISAL_Write when writing a sector to a disk.

The function can be addressed to a resource of any filter type, but that resource must not already be open by the device driver or by another filter.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Serial Number			
	Reserved = 0			
20	Logical Block Address			
24	Reserved = 0		Flags	Priority
28 through n	Data			

Function This is the function code, 28, for FC_Write

Filter Type This is the filter type to which the function is directed.

Serial Number This is the serial number of the resource to be written.

Logical Block Address
This is the logical block address of the block to be written.

Flags This field controls the type of write to be executed. These are defined in detail in "FN_ISAL_Write" on page 185.

Priority	This field is reserved for future use.
Data	The data to be written. The number of bytes should be that of the blocksize of the disk. Only 512-byte blocksize is supported at this time.

Status_DDR No status data is returned for this function, but a Status_DDR that points to the following data is returned when result is AS_Success:

Byte	3	2	1	0
0	Reserved = 0			
4	Length = 0			

Result The following result fields can be returned:

AS_Success

AE_NotInTable	(Filter not FT_PhysicalDisk or serial number not found)
Illegal Request (range)	
AE_PhysWrapped	(Device is currently wrapped)
AE_PhysFormatting	(Device is currently formatting)
AE_PhysCertifying	(Device is currently certifying)
AE_PhysIniting	(Device is currently being initialized)
AE_Failure	(Unable to begin writing)
AE_AccessDenied	(Copied from FN_ISALMgr_Open)
AE_InvalidRID	(Copied from FN_ISALMgr_Open)
AE_LogOpen	(Copied from FN_ISALMgr_Open)
AE_ReservationConflict	(Copied from FN_ISAL_Write)
AE_HardwareError	(Copied from FN_ISAL_Write)
AE_NotReady	(Copied from FN_ISAL_Write)
AE_MediumError	(Copied from FN_ISAL_Write)
AE_InvalidSignature	(Copied from FN_ISAL_Write)
AE_Offline	(Copied from FN_ISAL_Write)

AE_FencedOut	(Copied from FN_ISAL_Write)
AE_FormatDegraded	(Copied from FN_ISAL_Write)
AE_FormatinProgress	(Copied from FN_ISAL_Write)
AE_WriteProtect	(Copied from FN_ISAL_Write)
AS_Warning	(Copied from FN_ISAL_Write)

FC_AdapterSN

This function returns the serial number of the adapter. This is required by the PC configurator when private disks are used.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	

Function	This is the function code, 29, for FC_AdapterSN
Filter Type	This is the filter type to which the function is directed. This must be FT_Adapter, which is not really a filter.

Status_DDR This is a pointer to the buffer that receives the following data when result is AS_Success:

Byte	3	2	1	0
0	Reserved = 0			
4	Length = 8			
8 through 15	AdapterID			

Length	Number of bytes that follow in this field (8)
AdapterID	This 8–byte binary number uniquely identifies the adapter. It is the serial number of the adapter as reported in the VPD.

Result The following result fields can be returned:

AS_Success

AE_Failure

AE_UnknownFunction

FC_CacheFormat

This function zeroes all the data in the cache if the data has all been destaged to the disk drive. The function is provided for security purposes.

If the cache contains any data that has not been destaged to the disk drive, the transaction is rejected with AE_Failure. The function should only be sent to the fast write filter; it is rejected with AE_UnknownFunction by all other filters.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4	Reserved = 0			Flag

Function	This is the function code, 30, for FC_CacheFormat
Filter Type	This is the filter type to which the function is directed. This must be FT_FastWriteFilter.
Flags	FU_NewBat (bit 0) set to 1b when battery has been replaced to clear count of power-on hours.

Status_DDR No status data is required for this function, but a Status_DDR that points to the following data is returned when result is AS_Success:

Byte	3	2	1	0
0	Reserved = 0			
4	Length = 0			

Result The following result fields can be returned:

AS_Success	
AE_Failure	(Cache contains data not yet destaged to disk)
AE_UnknownFunction	(Filter not FT_FastWriteFilter but a known filter. This is always returned by the SSA 4-Port RAID Adapter)
AE_NotInTable	(Unknown filter)

FC_InitSurf

This function initializes all blocks on the disk (except the reserved area) to zeroes. AS_Success is returned if this is started successfully.

The array-configuration service issues an FN_ISALMgr_Open to the physical disk before returning the result field to the FC_InitSurf transaction. If the disk opens successfully, the array-configuration service constructs a record that tracks the serial number of the disk being zeroed and the progress of the initialization.

The array-configuration service then sends FN_ISAL_InitSurf to initialize selected blocks. If an FC_ResrcList or FC_ResrcView transaction is issued to the disk being initialized, the progress of the initialization is returned in the SNS_Percent field.

When initialization completes successfully, the handle is closed and the record of the disk serial number is removed. If initialization fails, the handle is closed and a record kept for that disk so that FS_ResrcInitFailed can be returned to a subsequent FC_ResrcList or FC_ResrcView transaction. This failure record persists until the adapter is re-booted or a wrap, format, or InitSurf is issued.

FC_InitSurf can only be addressed to a physical disk through the FT_PhysicalDisk filter.

If the cache contains any data that has not been destaged to the disk drive, the transaction is rejected with AE_Failure. The function should only be sent to the fast write filter; it is rejected with AE_UnknownFunction by all other filters.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Serial Number			
	Reserved = 0			
20	Blocksize			

Function This is the function code, 31, for FC_InitSurf

Filter Type This is the filter type to which the function is directed. This must be FT_PhysicalDisk which is not really a filter.

Serial Number This is the serial number of the device to be tested.

Blocksize
This is the number of bytes in each block. This must be a number supported by the disk drive.

Status_DDR No status data is required for this function, but a Status_DDR that points to the following data is returned when result is AS_Success:

Byte	3	2	1	0
0	Reserved = 0			
4	Length = 0			

Result The following result fields can be returned:

AS_Success

AE_NotInTable	(Filter not FT_PhysicalDisk or serial Number not found)
Illegal Request (range)	
AE_PhysWrapped	(Device is currently wrapped)
AE_PhysFormatting	(Device is currently formatting)
AE_PhysCertifying	(Device is currently certifying)
AE_PhysIniting	(Device is currently initialising)
AE_Failure	(Unable to begin certifying)
AE_AccessDenied	(Copied from FN_ISALMgr_Open)
AE_InvalidRID	(Copied from FN_ISALMgr_Open)
AE_NotSupported	(Reported if the disk has a blocksize above 744 bytes, if the transaction is directed to a disk that has block LRC protection but the adapter does not have hardware support for block LRC.)
AE_UnknownFunction	
AE_LogOpen	(Copied from FN_ISALMgr_Open)

FC_HotspareCfgStatus

This function returns a set of flags that indicate the status of the hot spare configuration. The purpose of this function is to provide a method for configurators to determine if the hot spare configuration is set up appropriately.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	

Function This is the function code, 32, for FC_HotspareCfgStatus

Filter Type This is the filter type to which the function is directed. This must be FT_HotspareDisk.

Status_DDR This is a pointer to the following data:

Byte	3	2	1	0
0	Unused			
4	Length			
8	Reserved = 0			Status

Length	Number of bytes that follow this field.
Status	These bit significant flags indicate the following status of the hot spare configuration when set to 1b:

HC_NoHotSpareProtection	There are arrays without hot spares.
HC_PoolShortConfig	There are hot spare pools with less hot spares than originally configured.
HC_PoolLessMin	There are hot spare pools with fewer hot spares than the specified minimum number.
HC_PoolOutOfSync	There is at leaset one hot spare pool that is not in sync.
HC_NotProtectingComp	There are hot spare pools that are not protecting members.
HC_UnpreferredExchange	There are array members that have been replaced with hot spares from a pool that is different from the specified pool.

Result The following result fields can be returned:

AS_Success

AE_UnknownFunction

FC_HotsparePoolList

This function returns the list of hot spare pool numbers that are in use and the associated SSA network for each pool. It is valid for the same pool number to exist on different SSA networks. These should be treated as separate pools.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	

Function	This is the function code, 33, for FC_HotsparePoolList
Filter Type	This is the filter type to which the function is directed. This must be FT_HotspareDisk.

Status_DDR This is a pointer to the following data:

Byte	3	2	1	0
0	Unused			
4	Length			
8	Reserved = 0		NetworkID(1)	Pool(1)
12	Reserved = 0		NetworkID(2)	Pool(2)
..	.		.	.
4n+4	Reserved = 0		NetworkID(n)	Pool(n)

Length Number of bytes that follow this field (256 byte maximum).

Pool This is the pool number. There can be up to 32 pools on each SSA network.

NetworkID This defines the SSA network that the hot spare pool is on. It can be:

NetworkA

NetworkB

Result The following result fields can be returned:

AS_Success

AE_UnknownFunction

FC_HotsparePoolView

This function returns information about a specified pool.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4	Reserved = 0		Network	Pool

Function This is the function code, 34, for FC_HotsparePoolView

Filter Type This is the filter type to which the function is directed. This must be FT_HotspareDisk.

Pool The pool number of the hot spare pool.

Network This defines the SSA network that the pool is on. it can be:

NetworkA

NetworkB

Status_DDR This is a pointer to the following data. It consists of a list of the hot spare serial numbers followed by the array serial number plus member serial number for each member of that array for each array that has been configured with hot spares in the specified pool. If the user is not interested in the serial numbers of the arrays and members configured, the size of the Status_DDR data can be truncated.

Byte	3	2	1	0
0	Unused			
4	Length			
8	Hot spares minimum	Hot spares configured	Components in pool	Hot spares in pool
12	Reserved = 0			Status
16 through 28	Flag	Hot Spare Serial Number (1)		
.				
16(n-1)+16 through 16(n-1)+28	Flag	Hot Spare Serial Number (n)		
16(n-1)+32 through 16(n-1)+44	Flag	Array Serial Number (1)		
16(n-1)+48 through 16(n-1)+60	Flag	Component Serial Number		
16(n-1)+64 through 16(n-1)+76	Flag	Component Serial Number		
.				
16(n-1) +16(m-1)+ 16(c-1)+16 through 16(n-1) +16(m-1)+ 16(c-1)+28	Flag	Array (m) Serial Number		
16(n-1) +16m+ 16(c-1)+16 through 16(n-1) +16m+ 16(c-1)+28	Flag	Component (c) Serial Number		

Length Number of bytes that follow this field.

Hotspares in pool
Number of hot spares in the pool as seen by the adapter.

Components in pool
Number of components in the pool as seen by the adapter

Hotspares configured
Number of hot spares that are configured to be in the pool.

Hotspares minimum
Minimum number of hot spares specified for the pool.

Status These bit significant flags indicate the following status of the hot spare configuration when set to 1b:

HC_NoHotSpareProtection
There are arrays without hot spares.

HC_PoolShortConfig
There are fewer hot spares than originaly configured.

HC_PoolLessMin
There are fewer hot spares than the specified minimum number.

HC_PoolOutOfSync
The pool is not in sync.

HC_NotProtectingComp
The hot spare pool does not protect any members.

HC_UnpreferredExchange
There are array members that have been replaced with hot spares from a pool that is different from the specified pool.

Hot Spare Serial Number
Serial number of hot spares in this pool

Array Serial Number
Serial number of the arrays configured for hot spares in this pool.

Array Component Serial Number
Serial number of array members configured for hot spares in this pool.

Serial numbers are listed by the first array serial number followed by its members serial numbers followed by the seconf array's serial number followed by its member's serial numbers and so on.

Flags This indicates the type of serial number and can be:

HT_HotSpare Serial number identifies a hot spare.

HT_Array Serial number identifies an array.

HT_Component
Serial number identifies an array member.

HT_TooLarge This indicates that there are hot spares in this pool that are too small for this member.

HT_WrongPool This indicates that this member was replaced with a hot spare from an unpreferred pool

Result The following results can be returned:

AS_Success

AE_UnknownFunction

AE_BadParameterValues

FC_ReadArrayHotspareParams

This function returns information about the hot spare configuration of each member of a specified array. This information can then be modified using FC_WriteArrayHotspareParams or FC_ModifyResrcParms. This transaction can also be used to determine whether the array has enough hot spare protection.

Parameter_DDR

This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Serial Number			
	Reserved = 0			

Function This is the function code, 35, for FC_ReadArrayHotspareParams.

Filter Type This is the filter type to which the function is directed.

Serial Number Serial number of the selected array.

Status_DDR This is a pointer to the following data:

Byte	3	2	1	0
0	Unused			
4	Length = 0			
8	Reserved = 0			Flags
12	Reserved = 0		Preferred (1)	Pool (1)
16	Reserved = 0		Preferred (2)	Pool (2)
.	.			
4n+8	Reserved = 0		Preferred (n)	Pool (n)

Length Number of bytes that follow this field

Flags These bit significant flags indicate the following configuration options of the hot spares when set to 1b:

HP_HotSpareEnabled
The array is configured to use hot spares.

HP_HotSpareExact
The array is configured to use hot spares of exactly the same size as its members.

HP_HotSparePreferred
The array is configured to be able to use hot spares only from the configured pool.

HP_ConfigOK The array is configured properly for hot spares.

Pool The hot spare pool number assigned to this member of the array. If a FT_3rdCopyFilter array is coupled to the array addressed in this transaction, the pool numbers for the FT_3rdCopyFilter members are returned after the pool numbers of the members of the addressed array.

Preferred The number of prefered hot spares that are suitable for this member of the array.

The first pool and preferred values are for the first member of the array, the second values are for the second component of the array and so on for all the array members.

Result The following result fields can be returned:

AS_Success

AE_UnknownFunction

AE_BadResrcSerialNumber

AE_Offline

FC_WriteArrayHotspareParams

This function updates the hot spare configuration for the specified array.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Array Serial Number			
	Reserved = 0			
20	Pool(4)	Pool(3)	Pool(2)	Pool(1)
.	.	.	.	.
20+(n/4)	.	.	.	Pool(n)

Function This is the function code, 36, for FC_WriteArrayHotspareParams.

Filter Type This is the filter type to which the function is directed.

Array Serial Number
Serial number of the selected array.

Pool Number of the pool for each member of the array. The length of DDR data must be even, so if the number of members is odd an extra byte should be supplied after the pool number of the last array member.

If an FT_3rdCopyFilter array is coupled to the array addressed in this transaction, the pool numbers for the FT_3rdCopyFilter members must follow the pool numbers of the members of the addressed array.

Status_DDR No status is returned for this function, but a status_DDR that points to the following data is returned when result is AS_Success:

Byte	3	2	1	0
0	Unused			
4	Length = 0			

Result The following result fields can be returned:

AS_Success

AE_UnknownFunction

AE_BadResrcSerialNumber

AE_Offline

AE_BadComponent

AE_BadParameterValues

AE_NotSupported
(This is reported if any adapter in the network does not permit this function)

FC_DeconfigureDisk

This transaction provides a mechanism for removing a member disk from an array. It was not supported on early releases of adapter firmware.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Serial Number			
	Reserved = 0			
20	Deconfigure Type			

Function This is the function code, 37, for FC_DeconfigureDisk.

Filter Type This is the filter type to which the function is directed. This can be:

FT_RAID1Filter

FT_RAID5Filter

FT_RAID10Filter

FT_HotSpareDisk

Serial Number Serial number of the selected disk.

Deconfigure Type
For FT_HotSpareDisk this field is ignored and the resource is disowned.

For RAID filters this field determines what conditions must be satisifed to perform the deconfigure:

0 Deconfigure the resource regardless of whether data loss will occur. Consider using a hotspare to replace the component. If no hotspare is available then degrade the array.

1 Only deconfigure the disk if data loss will not occur. Consider using a hotspare to replace the component. If no hotspare is available then degraded the array.

2 Only deconfigure the disk if data loss will not occur and there is a suitable hotspare to replace the component. Replace the component with the hotspare.

All other values are reserved

Status_DDR No status is returned for this function, but a status_DDR that points to the following data is returned when result is AS_Success:

Byte	3	2	1	0
0	Unused			
4	Length = 0			

Result The following result fields can be returned:

AS_Success The resource serial number was identified and the resource was successfully deconfigured.

AE_UnknownFunction
The transaction is not supported by this filter type

AE_BadParameterValues
The Deconfigure Type field was invalid

AE_BadComponentSerialNumber
The resource serial number is not recognised by this filter

AE_DataLossWillOccur
Returned by RAID filters when the deconfigure will cause data loss to occur and the parameter DDR states that data loss must not occur. Filters will return this error in preference to AE_NoHotspareAvailable if data loss will occur and there is no hotspare available.

AE_NoHotspareAvailable
Returned by RAID filters when the deconfigure will cause an array to become degraded because no hotspare is available and the parameter DDR states that a hotspare must be taken.

AE_BadComponent
The resource serial number was identified but the resource could not be successfully deconfigured

FC_CoupleArray

This transaction is used to couple a FT_RAID1Filter or FT_RAID10Filter array to a FT_3rdCopyFilter array. It is issued to the parent RAID filter, specifying the array to be coupled in the Parameter DDR. The Owning Module type of the FT_3rdCopyFilter must be of type OM_NotOwned. When this transaction returns AS_Success, the FT_3rdCopyFilter array is no longer be present in FC_ResrcList and the copy process is started.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16		Parent Array Serial Number		
	Reserved = 0			
20 through 36		To-be Coupled Serial Array Number		
	Reserved = 0			

Function This is the function code, 38, for FC_CoupleArray.

Filter Type This is the filter type to which the function is directed. This can be:

FT_RAID1Filter

FT_RAID10Filter

Parent Array Serial Number
Serial number of the array that the FT_3rdCopyFilter is to be coupled to.

To-be Coupled Serial Array Number
Serial number of the FT-3rdCopyFilter array that is to be coupled.

Status_DDR No status is returned for this function, but a status_DDR that points to the following data is returned when result is AS_Success:

Byte	3	2	1	0
0	Unused			
4	Length = 0			

Result The following result fields can be returned:

AS_Success The transaction was successful

AE_UnknownFunction
The filter does not understand the function code

AE_NotSupported
This can be returned if any adapter in the network does not permit this function.

AE_BadResrcSerialNumber
Either of the two serial numbers are not recognised or either array is not in the correct state to do this transaction.

AE_Offline The parent array is currently offline.

AE_CoupleOffline
The coupled array is currently offline.

FC_UncoupleArray

This transaction is used to decouple a FT_3rdCopy Filter array from its parent array. It is issued to the parent RAID filter such as FT_RAID1Filter or FT_RAID10Filter.

The Xmit DDR contains 128 bytes of host software specific data, exclusively for use by the host software. This buffer may contain anything the host software wishes. This 128 byte buffer can be used to store host software data such as the time of the decouple and the copy state. To retrieve this data, the FC_ReadUncoupledArrayMetaData transaction should be used. To update this data, the FC_WriteUncoupledMetaData transaction should be used.

The serial number of the uncoupled array is returned in the Status_DDR data. The uncoupled array has an Owning Module Type of OM_NotOwned.

In the event of a FT_3rdCopyFilter array being present, but the parent RAID-1/10 not present, a ′Phantom FS_ResrcUnknown′ array is generated so that the copy array can be deleted.

If the to-be uncoupled array is offline, AE_CoupleOffline is returned, unless the Destroy flag is set, in which case the two arrays will be uncoupled, then the uncoupled array will be destroyed. The components of the uncoupled array will be set back to owning module type of OM_NotOwned. This is regardless of whether the partner adapter understands 3-Way Copy or not.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Reserved = 0	Parent Array Serial Number		
20	Reserved = 0		Length	Destroy
24	128 byte maximum host software specific buffer			

Function This is the function code, 39, for FC_UncoupleArray.

Filter Type This is the filter type to which the function is directed. This can be:

FT_RAID1Filter

FT_RAID10Filter

Parent Array Serial Number
Serial number of the array that the FT_3rdCopyFilter is to be uncoupled from.

Destroy The Destroy flag causes the to-be uncoupled array to be destroyed and its components will then have an owning module type of OM_NotOwned

Length This is the length of the host software specific data that follows. This can be up to 128 bytes.

Software Specific Data
This data may be used by the host software for example to describe the uncoupled array or the progress of the host uncoupling.

Status_DDR This is a pointer to the following data when the result is AS_Success.

Byte	3	2	1	0
0	Unused			
4	Length = 16			
8 through 24	Reserved = 0	Uncoupled Array Serial Number		

Length Length of the following Status_DDR data (16 bytes)

Uncoupled Array serial Number
Serial number of the FT_3rdCopyFilter uncoupled array.

Result The following result fields can be returned:

AS_Success The transaction was successful

AE_UnknownFunction
The filter does not understand this function code

AE_Offline The parent array is currently offline. This is not a valid error code if the Destroy flag is set.

AE_BadResrcSerialNumber
The serial number is not recognised by this filter or the array does not have an array coupled to it.

AE_NotSupported
This is reported if any adapter in the network does not permit this function.

AE_IllReqShortDDR
The Transmit DDR length is too short.

AE_CoupleOffline
The to-be uncoupled array is offline and the Destroy flag was not set in the Parameter DDR.

AE_CoupleCopying
The to-be uncoupled array has not yet completed its copy. This will be returned if the Destroy flag is not set.

AE_CoupleDegraded
The to-be uncoupled array is degraded. This will be returned if the Destroy flag is not set

FC_ReadUncoupledMetaData

This transaction may be used to retrieve the meta-data stored by the user when FC_UncoupleArray was used. This transaction may only be used on uncoupled arrays that are listed in FC_ResrcList.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16		Array Serial Number		
	Reserved = 0			

Function This is the function code, 40, for FC_ReadUncoupledArrayMetaData.

Filter Type This is the filter type to which the function is directed. This must be:

FT_3rdCopyFilter

Array Serial Number
Serial number of the FT_3rdCopyFilter array.

Status_DDR This is a pointer to the following data when the result is AS_Success.

Byte	3	2	1	0
0	Unused			
4	Length = 16			
8 to 135 max	Host software specific data (128 bytes max)			

Length Length of the following Status_DDR data (128 bytes maximum)

Host Software Specific Data
Data originally written by a FC_UncoupleArray or later modified by a FC_WriteUncoupledMetaData

Result The following result fields can be returned:

AS_Success The transaction was successful

AE_UnknownFunction
The filter does not understand this function code

AE_Offline The parent array is currently offline.

AE_BadResrcSerialNumber
The filter does not recognise the specified serial number.

AE_IllReqShortDDR
The status DDR is too short for the metadata.

FC_WriteUncoupledMetaData

This transaction may be used to change the meta-data stored by the user when FC_UncoupleArray was used. This transaction may only be used on uncoupled arrays that are listed in FC_ResrcList.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 516	Array Serial Number			
	Reserved = 0			
20	Length			
24 to 141 max	Host software specific data (128 bytes maximum)			

Function This is the function code, 46, for FC_WriteUncoupledArrayMetaData.

Filter Type This is the filter type to which the function is directed. This must be:

FT_3rdCopyFilter

Array Serial Number
Serial number of the FT_3rdCopyFilter array.

Length Length of the following host software specific data (128 bytes maximum).

Host software specific data
Host software data that can be used to identify the uncoupled array and the progress of the host uncoupling.

Status_DDR This is a pointer to the following data when the result is AS_Success.

Byte	3	2	1	0
0	Unused			
4	Length = 16			

Length Length of the following Status_DDR data (128 bytes maximum)

Host Software Specific Buffer
Data originally written by a FC_UncoupleArray.

Result The following result fields can be returned:

AS_Success The transaction was successful

AE_UnknownFunction
The filter does not understand this function code

AE_Offline The uncoupled array is currently offline.

AE_BadResrcSerialNumber
The filter does not recognise the specified serial number.

FC_CoupleCompCandCount

This transaction is similar to FC_CompExchCandCount. It is issued to the parent FT_RAID1Filter or FT_RAID10Filter. It returns a count of OM_NotOwned candidates that are suitable for the parent array specified in the parameter DDR, taking into consideration the size of components, block size format and block LRC. The user may then issue an FC_ResrcCreate with these resources to create a FT_3rdCopyFilter and couple it to the chosen FT_RAID1Filter or FT_RAID10Filter array.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Array Serial Number			
	Reserved = 0			

Function This is the function code, 41, for FC_CoupleCompCandCount.

Filter Type This is the filter type to which the function is directed. This can be:

FT_RAID1Filter

FT_RAID10Filter

Array Serial Number
Serial number of the RAID-1 or RAID-10 array.

Status_DDR This is a pointer to the following data when the result is AS_Success.

Byte	3	2	1	0
0	Unused			
4	Length = 4			
8	Count			

Length Length of the following Status_DDR data (4 bytes)

Count Number of OM_NotOwned components that are candidates for a FT_3rdCopyFilter array for the specified array.

Result The following result fields can be returned:

AS_Success The transaction was successful

AE_UnknownFunction
The filter does not understand this function code

AE_BadResrcSerialNumber
The filter does not recognise the array serial number.

AE_ArrayIsBroken
The array is currently offline or unknown

AE_Failure An internal transaction error occurred.

FC_CoupleCompCandList

This transaction is similar to FC_CompExchCandList. It is issued to the parent FT_RAID1Filter or FT_RAID10Filter to return a list of OM_NotOwned candidates that are suitable for the parent array specified in the parameter DDR taking into consideration the size of components, block size format and block LRC. The user may then issue a FC_ResrcCreate with these resources to create their FT_3rdCopyFilter and couple the chosen FT_RAID1Filter or FT_RAID10Filter array.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16		Array Serial Number		
	Reserved = 0			

Function This is the function code, 42, for FC_CoupleCompCandList.

Filter Type This is the filter type to which the function is directed. This can be:

FT_RAID1Filter

FT_RAID10Filter

Array Serial Number
Serial number of the RAID-1 or RAID-10 array.

Status_DDR This is a pointer to the following data when the result is AS_Success.

Byte	3	2	1	0
0	Unused			
4	Length			
8 through 20		Serial Number (n)		
	Reserved = 0			
24	Reserved = 0		Percent	Status
28 through 40		Serial Number (n + 1)		
	Reserved = 0			
44	Reserved = 0		Percent	Status
. .				
20m-12 through 20m		Serial Number (n + m)		
	Reserved = 0			
20m+4	Reserved = 0		Percent	Status

Length Length identifies the number of bytes that follow this field.

Serial Number
This is the serial number of each candidate.

Status This can be one of the following:

FS_CandOnline
The component is in the RS_Online state.

FS_CandOffline
The component is in the RS_Offline state.

Percent
This field is zero.

Result The following result fields can be returned:

AS_Success The transaction was successful

AE_UnknownFunction
The filter does not understand this function code

AE_BadResrcSerialNumber
The filter does not recognise the array serial number.

AE_ArrayIsBroken
The array is currently offline or unknown

AE_Failure An internal transaction error occurred.

FC_CoupleResrcCandCount

This transaction is to be issued to the parent FT_RAID1Filter or FT_RAID10Filter. It returns a count of the number of uncoupled FT_3rdCopyFilter array's suitable for the parent array as specified in the Parameter DDR. This transaction takes into consideration the number of members in the array, the minimum size a member must be for the array, formatted block size of each member and block LRC.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Array Serial Number			
	Reserved = 0			

Function This is the function code, 43, for FC_CoupleResrcCandCount.

Filter Type This is the filter type to which the function is directed. This can be:

FT_RAID1Filter

FT_RAID10Filter

Array Serial Number
Serial number of the RAID-1 or RAID-10 array.

Status_DDR This is a pointer to the following data when the result is AS_Success.

Byte	3	2	1	0
0	Unused			
4	Length = 4			
8	Count			

Length Length of the following Status_DDR data (4).

Count Number of FT_3rdCopyFilter arrays that are candidates for a FT_3rdCopyFilter array for the specified array.

Result The following result fields can be returned:

AS_Success The transaction was successful

AE_UnknownFunction
The filter does not understand this function code

AE_BadResrcSerialNumber
The filter does not recognise the array serial number.

AE_ArrayIsBroken
The array is currently offline or unknown

AE_Failure An internal transaction error occurred.

FC_CoupleResrcCandList

This transaction is to be issued to the parent FT_RAID1Filter or FT_RAID10Filter. It returns a list of uncoupled FT_3rdCopyFilter array's suitable for the parent array specified in the parameter DDR. This transaction takes into consideration the number of components in the array, the minimum size a member must be for the array, formatted block size of each component and block LRC.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Array Serial Number			
	Reserved = 0			

Function This is the function code, 44, for FC_CoupleResrcCandList.

Filter Type This is the filter type to which the function is directed. This can be:

FT_RAID1Filter

FT_RAID10Filter

Array Serial Number
Serial number of the RAID-1 or RAID-10 array.

Status_DDR This is a pointer to the following data when the result is AS_Success.

Byte	3	2	1	0
0	Unused			
4	Length			
8 through 20	Reserved = 0	Serial Number (n)		
24	Reserved = 0		Percent	Status
28 through 40	Reserved = 0	Serial Number (n + 1)		
44	Reserved = 0		Percent	Status
. .				
20m-12 through 20m	Reserved = 0	Serial Number (n + m)		
20m+4	Reserved = 0		Percent	Status

Length Length identifies the number of bytes that follow this field

Serial Number
This is the serial number of each candidate

Status This can be one of the following:

FS_CandOnline
The component is in the RS_Online state.

FS_CandOffline
The component is in the RS_Offline state.

Percent
This field is zero.

Result The following result fields can be returned:

AS_Success The transaction was successful

AE_UnknownFunction
The filter does not understand this function code

AE_BadResrcSerialNumber
The filter does not recognise the array serial number.

AE_ArrayIsBroken
The array is currently offline or unknown

AE_Failure	An internal transaction error occurred.

FC_CoupledArrayComponentView

This transaction is similar in usage to FC_ComponentView and may only be used on a FT_RAID1Filter or FT_RAID10Filter array. If there is an array coupled with the specified array in the parameter DDR, this transaction lists the additional components and their status, similar to FC_ComponentView.

If there is not an array coupled, the Status DDR will not contain any components and the length will be adjusted accordingly.

Parameter_DDR
This is a pointer to the following data:

Byte	3	2	1	0
0	Reserved = 0	Filter Type	Function	
4 through 16	Serial Number			
	Reserved = 0			
20	First Component (n)			
24	Requested Component (m)			

Function
This is the function code, 45, for FC_CoupledArrayComponentView.

Filter Type
This is the filter type to which the function is directed.

Resource Serial Number
This is the name of the array. It is a 15 character ASCII string.

First Component
This is the ordinal number of the first component (starting at zero) to be reported.

Requested Count
This is the maximum number of components that should be reported starting from the identified First Component.

Status_DDR This is a pointer to the following data when the result is AS_Success.

Byte	3	2	1	0
0	Unused			
4	Length			
8 through 20	Serial Number (n)			
	Reserved = 0			
24	Reserved = 0		Percent	Status
28 through 40	Serial Number (n + 1)			
	Reserved = 0			
44	Reserved = 0		Percent	Status
. .				
20(n+m)-12 through 20(n+m)	Serial Number (n + m)			
	Reserved = 0			
20(n+m)+4	Reserved = 0		Percent	Status

Length Length identifies the number of data bytes that follow this field

Serial Number
This is the serial number of each candidate. It is a 15 character ASCII string.

Status This can be one of the following:

FS_CompPresent

FS_CompNotPresent

FS_CompNotPresentBlank
Component has been deconfigured from the array.

FS_CompIllegalNetwork
This is returned when components of an array are not all on the same SSA loop. If there is a single component on a different loop from the others, FS_CompIllegalNetwork is returned only for that component. If more than one component are on a different loop, FS_CompIllegalNetwork is returned for all components of the array.

Percent
This field is zero.

Result The following result fields can be returned:

AS_Success The transaction was successful

AE_UnknownFunction
The filter does not understand this function code

AE_BadResrcSerialNumber	The filter does not recognise the array serial number.

System Transactions

Result	The following result fields can be returned:

AS_Success

AE_Failure – Cache contains data not yet destaged to disk

AE_UnknownFunction – Filter not FT_FastWriteFilter but a known filter

AE_NotInTable – Unknown filter

All services support the standard IPN System transactions as follows:

Transaction	Minor_function
SF_Stop	1
SF_Ping	2
SF_Finger	3
SF_Powerfailure	4
SF_Version	5

SF_Stop

This transaction is sent to a service by IPN to close the service down in an orderly way.

Minor_function	1
Parameter_DDR	Null
Transmit_DDR	Null
Receive_DDR	Null
Status_DDR	Null

SF_Ping

Ping is like the TCP/IP function of the same name. It simply returns AS_Success if the slave service is running. This says nothing about the health of the slave application. A disk service with a broken disk drive would still return a good result.

Minor_function	2
Parameter_DDR	Null
Transmit_DDR	Null
Receive_DDR	Null
Status_DDR	Null

SF_Finger

SF_Finger is a reserved function.

Minor_function	3
Parameter_DDR	Null
Transmit_DDR	Null
Receive_DDR	Null
Status_DDR	Null

SF_PowerFailure

This transaction is sent to all services by IPN when the node power may be about to fail. A slave service should save any vital information but remain in a state to process new transactions should the power recover.

Minor_function	4
Parameter_DDR	Null
Transmit_DDR	Null
Receive_DDR	Null
Status_DDR	Null

SF_Version

This transaction can be sent to a service to obtain its code level.

Minor_function	5
Parameter_DDR	Null
Transmit_DDR	Null
Receive_DDR	Null
Status_DDR	Points to the buffer allocated to receive the Version level. The version is a 32 bit unsigned integer that indicates the current level of the code for the service. If SF_Version is not supported, an application result AE_UnknownFunction is returned.

Application Results

The following Application-result fields might be returned at the end of a transaction:

AS_Success	The transaction has been successfully completed.
AS_Warning	The transaction has been successfully completed, but the Status DDR contains warning information that either the unit might fail soon or a logical block should be reassigned.
AE_NotReady	The service is not ready to execute this transaction.

AE_MediumError
The transaction has terminated with a nonrecoverable error condition caused by a flaw in the disk surface or an error in the recovered data. The Status DDR contains the address of the logical block in error.

AE_HardwareError
A nonrecoverable hardware error was detected during this transaction.

For RAID arrays there are two reasons why this can occur:

1. A DMA operation between the host and the adapter failed (excludes sequence number and LRC failures).
2. A read/write operation fails because of two or more member failures. For example a write to a rebuilding array will be failed with AE_HardwareError if a member write to the non-rebuilding member fails.

Note: Normally a media error is not counted as a member failure, however there are some scenarios in RAID-5 where the combination of a media error and a disk failure can cause I/O to be failed with AE_HardwareError.

Note: There is also a RAID-5 scenario where two media errors on different members but the same member LBA can cause AE_HardwareError to be returned. This scenario requires read I/Os to be submitted for both sectors at the same time.

AE_ReservationConflict
The transaction was not executed because the resource was reserved to another client.

AE_WriteProtect
The transaction was not executed because write operations are not permitted to the resource.

AE_Failure The transaction could not be completed for a reason other than an error.

AE_AccessDenied
Access is denied because of the mode in the open operation for that resource.

Illegal Request (range)
There was an illegal field in the transaction. A range of result field codes are reserved for Illegal Request to provide more information on the field in error.

AE_Offline The resource was in the RS_Offline state and the transaction to this handle could not be executed. The only valid transactions that can be addressed to a handle for a resource in the RS_Offline state are FN_ISAL_Close, FN_ISALMgr_Characteristics, and FM_ISALMgr_Statistics.

AE_SCSIError Nonzero SSA-SCSI status was returned from the resource while opened in MD_SCSI mode.

AE_UnknownFunction	The function requested is not supported.
AE_LogOpen	This function cannot be executed while the corresponding logical resource is open. This could be an attempt to open the physical resource in MD_Service mode or to send certain transactions to the physical resource.
AE_SSAString	An attempt was made to open a physical resource in MD_Service mode while that resource was in an SSA string network rather than in a loop.
AE_FencedOut	The resource is currently fenced out from executing this transaction from this client.
AE_TableFull	Resource table is full.
AE_InvalidRID	Resource is currently not known by the recipient.
AE_NotInTable	Only returned to transactions originating from the registry.
AE_NotYetImplemented	Function is not yet implemented.
AE_RetryWhenMemory	Only returned to transactions originating from the registry.
AE_ClusterNumberNotKnown	Only returned in the FN_ISAL_Fence transaction.
AE_FormatDegraded	A format operation to the disk drive has unsuccessfully completed and the user data area is not accessible.
AE_FormatInProgress	The disk drive is currently executing a formatting operation.
AE_MissingCluster	The cluster number is not known to the system.
AE_RoutingError	Error in executing a TargetTransfer transaction to another node.
AE_RemoteTimeout	The remote host did not respond to a TransferToHost transaction within the timeout period.
AE_TargetNotAvailable	The remote node is not available to receive data from the sending cluster.
AE_TargetReceiverFull	The buffer in the remote host is not available to receive data.
AE_TargetTransferTooLarge	The buffer in the remote host is too small to receive the data specified.

AE_MediaReadOnly
Write operations are not permitted to this resource, for example, to a degraded array.

AE_ParityNotValid
Parity is not valid for an array.

AE_QNTimedOut
Returned only to FN_ADAP_QueryNodes to indicate that the node did not respond to the Query Node SMS that was sent.

AE_InConfig The adapter was in the process of SSA reconfiguration when the transaction was received.

AE_InServiceMode
The transaction cannot be executed because the resource is in service mode.

AE_OfflineTimeout

AE_NotFound

AE_PhysWrapped

AE_PhysCertifying

AE_PhysIniting

AE_PhysFormatting

AE_NotOpen

AE_TransferFailed

AE_TabAborted

AE_NonIsal

AE_NotSupported

AE_OtherAdapterInServiceMode

The following Application_result fields might be returned during configuration or array transactions:

AE_BadSequenceNumber

AE_BadTransferLRC

AE_BlockLRC

AE_BadSerialNumber

AE_BadOldSerialNumber

AE_BadNewSerialNumber

AE_BadComponentCount

AE_BadComponentSerialNumber

AE_BadResrcSerialNumber

AE_BadOldComponentSerialNumber

AE_BadNewComponentSerialNumber

AE_BadParameterValues

AE_ArrayIsBroken

AE_SetOMTFailed

AE_BadExchangeCandidate

AE_FiltersOnly

AE_NotFilters

AE_NvramError

AE_InvalidCandidateRequest

AE_InvalidCreateRequest

AE_ReadOnlyParameterValue

AE_ArrayIsBrokenOrDegraded

AE_AvoidWrite

AE_AvoidReadWrite

AE_NotLocal

AE_Flush CompFailure

AE_ConfirmRequired

AE_InvalidMetaRequest

AE_PhysIniting

Chapter 7. Error Recovery and Error Logging

This section defines the error recovery and error reporting that is performed by the adapter when adapter, SSA-link, or attached-device errors are detected.

Strategy

The following strategy is implemented for error recovery and error reporting:

Error Recovery

- All possible error recovery for the attached devices is performed by the adapter. The error recovery is based on async-alert conditions, SCSI status, or a decode of the SCSI Key/Code/Qualifier sense data. If an error log entry is to be made as a result of the error recovery procedure (ERP), the adapter sends the error code to the error logger with the ID of the failing physical resource and the ID of the error log template.
- SSA Link errors are recovered in accordance with the SSA link ERP specification included in the SSA specification. If the LINK ERP fails, the error data is logged against the resource ID of the adapter.
- The recovery procedures and error codes logged in the case of errors detected by the device driver that cause communication with the adaptor card to fail are defined in the device driver specifications.

Error Logging

The local adapter logs disk errors against the failing physical device. It logs all other errors against the adaptor. The remote adaptor in an N-way system (where N > 2) logs all transaction failures against the failing logical disk. (An hdisk on a pSeries, RS/6000, or SP/2 server.)

Errors are logged as a result of:
- Device errors reported by an I/O device
- Adapter-detected failures (These include errors in the adapter, arrays, SSA links, and SSA configuration.)
- Device-driver-detected failures
- Device driver healthcheck-detected errors

The following error data is logged:

- Error data logged against disk drives consists of the 32 bytes of SCSI sense data returned by the disk drive.
- Error data logged as a result of adapter-detected failures consists of up to 156 bytes of data. The first three bytes are an adapter error code. The remainder of the data depends on the error type. For array errors this may include the Array Name (Serial Number / Connection Address) of the array and also the unique ID (Serial number) of the failing component.

Error Record Templates

Each error code is assigned to an error log template.

The following is a list of error templates with their ID numbers. The ID numbers are used in byte 1 of the FN_REGY_LogErrorFromRegisty transaction to identify the template. The device driver translates the error-log-template ID to the system error ID.

01h	SSA_DISK_ERR4
02h	DISK_ERR4
03h	SSA_DISK_ERR2
04h	DISK_ERR1
05h	SSA_DISK_ERR3
06h	SSA_LINK_ERROR
07h	SSA_DETECTED_ERROR
08h	SSA_DEVICE_ERROR
09h	SSA_DEGRADED_ERROR
0Ah	SSA_HDW_ERROR
0Bh	SSA_HDW_RECOVERED
0Ch	SSA_SOFTWARE_ERROR
0Dh	SSA_LINK_OPEN
0Eh	SSA_DISK_ERR1
0Fh	SSA_LOGGING_ERROR
10h	SSA_ARRAY_ERROR
11h	SSA_SETUP_ERROR
13h	SSA_CACHE_ERROR
14h	SSA_CONFIG_COMPLETE
15h	SSA_CACHE_BATTERY
16h	SSA_ENCL_ERR1

Health Check Monitoring

Every hour the device driver requests that a health check is performed by the adapter. This results in error logs being generated for every error condition that exists within the adapter, either SSA loop, or within any disk drive. Any error reported following a health check would have been previously reported when the error was first detected. These errors may not prevent operation of commands, for example loss of redundant power condition. Reporting these errors every time a health check is issued ensures a sevice action will be taken even if the original error log report is not actioned or has been lost.

Device Error Recovery

Any device attached to the adapter reports failures by means of SCSI status codes, and SCSI sense data. For each device type attached to the adapter, a table is maintained in the adapter that defines the error recovery procedure (ERP) to be used and the data to log for all failure conditions reported by means of SCSI sense data. The default ERP table is for the SSA disk drives supplied with the 7133 SSA Subsystem. These ERP tables are built from data provided by the attaching devices. The data provided by the devices is in the following format:

Description	A text description of the error condition. This is here for information only and is not included in the ERP tables in the adapter microcode.
SCSI K/C/Q	The SCSI sense data key, code, and qualifier fields.
ERP#	The error recovery procedure to be used.
Log	The error-logging strategy that is used by the error recovery procedure as follows: 0 = No log entry 1 = Log the sense data 2 = Log first sense data if the ERP fails 3 = Log last sense data if the ERP fails
Template	The error logging template that should be used to log this error.

Bad Block Management

When a block cannot be read from a disk, but that data can be reconstructed from the other components of an array, the bad block is reassigned by a facility called *IDISK*.

IDISK notes the addresses of all blocks that return an unrecoverable medium error. When a write operation is next directed to such a block, IDISK reassigns that block before writing the new data. If the block cannot be reassigned by the drive, IDISK performs a software reassignment to an area of the disk outside the customer data area. Future read and write operations to that logical block address use the new block.

When a disk reports that it has an unrecoverable data error, the RAID-5 and RAID-1 filters rewrite the bad block when they have to reconstruct the lost data. The rewriting action causes IDISK to reassign the block before the data is rewritten.

SSA Link Error Recovery

The SSA link error recovery procedures are defined in the SSA functional specification. This section defines the error logging strategy that is applied when link errors are reported to the adapter by means of asynchronous alert codes.

When an asynchronous alert is received, the adapter that is the SSA master logs the error code in accordance with the adapter error logging data table below.

If the error recovery fails, all adapters on the network are left with an open loop. Under these circumstances, the adapters log an error code indicating that the serial link is in degraded mode.

For alerts of type 6, no error recovery is applied. However, if the asynchronous alert type is 06 and the subtype is 01 (redundant power failure), the adapter waits for a period to see if asynchronous alerts with the same type and subtype fields are reported from more than one device and then logs the appropriate error code as defined in the table below.

Adapter Error Logging Data

The following table defines the error codes and error-log templates used when adapter error recovery procedures are invoked. These are hexadecimal characters. The first digit of the error code is the threshold value that must be exceeded before an SRN is generated. Each of the errors in this table are logged against the adapter resource ID.

Table 61. Adapter Error Logging Data

Condition	**Template**	**Error Code**
No error (only returned for AdapterHealthCheck)	-	00 00 00
Async type = 00 - 01 (no log)	-	-
Async type = 02 Unknown message	SSA_LINK_ERROR	32 A0 02
Async type = 03 Invalid message	SSA_LINK_ERROR	32 A0 03
Async type = 04 Protocol error	SSA_LINK_ERROR	32 A0 04
Async type = 05 Environmental error (not reported)	SSA_DETECTED_ERROR	02 A0 05
Async type = 06, Subtype = 01. Where ERP finds only one async of this type and subtype	SSA_DETECTED_ERROR	02 A0 06
Async type = 06, Subtype = 01. Where ERP finds more than one device reporting this async type and subtype	SSA_DETECTED_ERROR	02 A1 06
Async type = 06 Subttype = 03. Port not operational and POSTs failed	SSA_DEVICE_ERROR	02 A2 06
Async type = 10 Permanent line fault P=port(0-3) HH=hop(00-99)	SSA_LINK_OPEN	22 0P HH
Async type = 11 No characters received P=port(0-3) HH=hop(00-99)	SSA_LINK_ERROR	A2 1P HH
Async type = 12 Remote port disabled P=port(0-3) HH=hop(00-99)	SSA_LINK_ERROR	A2 2P HH

Table 61. Adapter Error Logging Data (continued)

Condition	Template	Error Code
Async type = 13 Link reset failed P=port(0-3) HH=hop(00-99)	SSA_LINK_ERROR	A2 3P HH
Async type = 14 Retry limit exceeded P=port(0-3) HH=hop(00-99)	SSA_LINK_ERROR	02 4P HH
Async type = 15 Hardware error P=port(0-3) HH=hop(00-99)	SSA_LINK_ERROR	A2 5P HH
Async type = 16 Frame reject P=port(0-3) HH=hop(00-99)	SSA_LINK_ERROR	A2 6P HH
Async type = 17 Invalid retry status P=port(0-3) HH=hop(00-99)	SSA_LINK_ERROR	A2 7P HH
Async type = 18 Time-out waiting for Disabled state P=port(0-3) HH=hop(00-99)	SSA_LINK_ERROR	A2 8P HH
Async type = 19 Time-out waiting for Ready state P=port(0-3) HH=hop(00-99)	SSA_LINK_ERROR	A2 9P HH
Invalid async. code	SSA_LINK_ERROR	32 FF FF
Excessive link reconfiguration detected	SSA_LINK_ERROR	03 3P HH
Single device reports loss of redundant power, cooling, over temperature, or drawer bypass card failure	SSA_DETECTED_ERROR	03 00 C0
Multiple devices report loss of redundant power, cooling, over temperature, or drawer bypass card failure	SSA_DETECTED_ERROR	03 01 C0
Incorrect length of data transferred by the device, or Block LRC error on Read	SSA_DEVICE_ERROR	03 03 FE
Invalid SCSI Status received	SSA_DEVICE_ERROR	03 03 FF
Reallocations exceeded	SSA_DEVICE_ERROR	03 10 00
Adapter issued Device Reset message	SSA_DEVICE_ERROR	23 10 00
Adapter Hardware Failure	SSA_HDW_ERROR	04 00 00
Adapter 64 MB SDRAM module failure	SSA_DEGRADED_ERROR	04 00 64
Adapter 128 MB SDRAM module failure	SSA_DEGRADED_ERROR	04 01 28
No SDRAN installed, or POST unable to determine SDRAM size	SSA_HDW_ERROR	04 20 00
Other adaptors in SSA network are using incompatible microcode levels	SSA_DEGRADED_ERROR	04 22 00
Adapter Fast Write Cache card failure (No data loss)	SSA_DEGRADED_ERROR	04 25 00
Not enough SDRAM for Fast Write Cache	SSA_DEGRADED_ERROR	04 25 10
Fast Write resource detected, but no Fast Write Cache card on adapter	SSA_DEGRADED_ERROR	04 25 15
Incorrect data on identified resource (LBAs not known) - Data loss	SSA_CACHE_ERROR	04 25 21
Bad version number of Fast Write Cache	SSA_CACHE_ERROR	04 25 23
Fast Write Cache not accessible for the resource	SSA_CACHE_ERROR	04 25 24
Fast Write Cache is not correct for the resource	SSA_CACHE_ERROR	04 25 25
Dormant Fast Write cache entry exists	SSA_DEGRADED_ERROR	04 25 27
Duplicate fast Write disk serial number detected	SSA_DEGRADED_ERROR	04 25 28

Table 61. Adapter Error Logging Data (continued)

Condition	**Template**	**Error Code**
Cache disabled - battery charging (Normal state after power-on)	SSA_DEGRADED_ERROR	04 25 29
System voltage low - cache switched to self-refresh	SSA_DEGRADED_ERROR	24 25 2A
Battery failure - cache disabled	SSA_CACHE_BATTERY	04 25 2B
Battery requires replacement	SSA_CACHE_BATTERY	04 25 2C
Fast Write Caching is suspended for one or more devices	SSA_DEGRADED_ERROR	04 25 2D
2–way Fast Write Cache is configured to only operate when both caches are available but one cache is now unavailable	SSA_DEGRADED_ERROR	04 25 40
SSA device is preventing the completion of link configuration, where P=port (0-3) and HH=decimal hop (00-99) of the failing device	SSA_DEVICE_ERROR	04 3P HH
Device with ’failed’ status where P=port (0-3) and HH=decimal hop (00-99) of the failed device	SSA_DEVICE_ERROR	04 4P HH
Open SSA Link where P=port (0-3) and HH=decimal hop (00-99) of the first device that is not accessible on the shortest link	SSA_LINK_OPEN	24 5P HH
Array offline - more than one disk not available in RAID-0 or more than two disks not available in RAID-5	SSA_DEGRADED_ERROR	04 60 00
Array component missing or remote NVRAM not available	SSA_DEGRADED_ERROR	04 65 00
Minimum adapter resources not available for array filter	SSA_ARRAY_ERROR	04 70 00
Incorrect data area of array	SSA_ARRAY_ERROR	04 75 00
Illegal Link Configuration (SIC-SIC/>48 drives/>1 adapter)	SSA_DEGRADED_ERROR	04 80 00
Illegal Link Configuration detected by array filter	SSA_DEGRADED_ERROR	04 85 00
One array member disk is on a different loop	SSA_DEGRADED_ERROR	04 86 00
Multiple array member disks are on a different loop	SSA_DEGRADED_ERROR	04 87 00
Array offline because the primary or secondary half of the array is not present	SSA_DEGRADED_ERROR	04 87 50
Array offline because the adapter is unknown to the remaining half of the array	SSA_DEGRADED_ERROR	04 87 55
Array offline because split/join procedure was not performed correctly	SSA_DEGRADED_ERROR	04 87 60
Array not available - invalid strip table full	SSA_ARRAY_ERROR	04 88 00
Array not available - multiple device failures	SSA_ARRAY_ERROR	04 89 00
Array degraded. (Component unconfigured during initial parity build)	SSA_ARRAY_ERROR	04 89 50
Array degraded - one disk not available	SSA_ARRAY_ERROR	04 90 00
Array exposed - one disk not available	SSA_ARRAY_ERROR	04 91 00
No spares available for an array that is configured for spares	SSA_ARRAY_ERROR	04 95 00
Hot spares in the same pool have different views of the hot spare configuration	SSA_SETUP_ERROR	04 95 10

Table 61. Adapter Error Logging Data (continued)

Condition	Template	Error Code
An array member has used a hot spare from a pool other than its specified pool	SSA_DEGRADED_ERROR	04 95 20
The minimum number of hot spares in the assigned pool is greater than the number of hot spares currently in the pool	SSA_DEGRADED_ERROR	04 95 30
Hot spares have been assigned to pools other than pool zero but other adapters on the SSA loop are using versions of microcode that do not support spare pools	SSA_DEGRADED_ERROR	04 95 40
Incorrect parity in array	SSA_ARRAY_ERROR	04 97 00
Different adapter on each loop	SSA_DEGRADED_ERROR	04 98 00
Array copy is degraded	SSA_DEGRADED_ERROR	04 99 50
Adapter is unable to initialize a device	SSA_DEVICE_ERROR	04 A1 00
Unable to configure device where P=port (0-3) and HH=decimal hop (00-99) of the device	SSA_DEVICE_ERROR	04 BP HH

SSA Disk Drive Error Recovery Table

The following are the error recovery procedures implemented in the adapter for SSA disk drives when connected to a SSA adapter. Each of the errors in this table is logged against the disk drive resource ID.

Table 62. SCSI Sense Key/Code/Qualifier recovery procedures

Description	SCSI K/C/Q	ERP#	Log	Template
No Additional Sense Information	0 00 00	1	0	-
No Index/Sector Signal	1 01 00	1	1	SSA_DISK_ERR2
No Seek Complete	1 02 00	1	1	SSA_DISK_ERR2
Peripheral Device Write Fault	1 03 00	1	1	SSA_DISK_ERR2
Track Following Error	1 09 00	1	1	SSA_DISK_ERR2
Temperature Warning Error	1 0B 01	1	1	SSA_DISK_ERR4
Write Error Recovered With Auto Reallocation	1 0C 01	1	1	SSA_DISK_ERR3
Write Error - Recommend Reassignment	1 0C 03	2	1	SSA_DISK_ERR3
Record Not Found	1 14 01	1	1	SSA_DISK_ERR3
Record Not Found - Recommend Reassignment	1 14 05	2	1	SSA_DISK_ERR3
Record Not Found - Data Auto Reallocated	1 14 06	1	1	SSA_DISK_ERR3
Random Positioning Error	1 15 00	1	1	SSA_DISK_ERR2
Positioning Error Detected by Read of Medium	1 15 02	1	1	SSA_DISK_ERR2
Data Synchronization Mark Error	1 16 00	1	1	SSA_DISK_ERR3
Data Synchronization Mark Error - Data Rewritten	1 16 01	1	1	SSA_DISK_ERR3
Data Synchronization Mark Error - Recommend Rewrite	1 16 02	3	1	SSA_DISK_ERR3

Table 62. SCSI Sense Key/Code/Qualifier recovery procedures (continued)

Description	SCSI K/C/Q	ERP#	Log	Template
Data Synchronization Mark Error - Data Auto-Reallocated	1 16 03	1	1	SSA_DISK_ERR3
Data Synchronization Mark Error - Recommend Reassignment	1 16 04	2	1	SSA_DISK_ERR3
Recovered Data With Retries	1 17 01	1	1	SSA_DISK_ERR3
Recovered Data with Positive Head Offset	1 17 02	1	1	SSA_DISK_ERR3
Recovered Data with Negative Head Offset	1 17 03	1	1	SSA_DISK_ERR3
Recovered Data using previous sector ID	1 17 05	2	1	SSA_DISK_ERR3
Recovered Data Without ECC - Data Auto-Reallocated	1 17 06	1	1	SSA_DISK_ERR3
Recovered Data Without ECC - Recommend Reassignment	1 17 07	2	1	SSA_DISK_ERR3
Recovered Data Without ECC - Recommend Rewrite	1 17 08	3	1	SSA_DISK_ERR3
Recovered Data Without ECC - Data Rewritten	1 17 09	1	1	SSA_DISK_ERR3
Recovered Data with Error Correction and Retries Applied	1 18 01	1	1	SSA_DISK_ERR3
Recovered Data - Data Auto-Reallocated	1 18 02	1	1	SSA_DISK_ERR3
Recovered Data - Recommend-Reassignment	1 18 05	2	1	SSA_DISK_ERR3
Recovered Data With ECC - Recommend Rewrite	1 18 06	3	1	SSA_DISK_ERR3
Recovered Data With ECC - Data Rewritten	1 18 07	1	1	SSA_DISK_ERR3
Primary Defect List Not Found	1 1C 01	12	1	SSA_DISK_ERR2
Grown Defect List Not Found	1 1C 02	12	1	SSA_DISK_ERR2
Partial Defect List Transferred	1 1F 00	12	1	SSA_DISK_ERR2
Internal Target Failure	1 44 00	1	1	SSA_DISK_ERR2
Spindles Not Synchronized	1 5C 02	15	2	SSA_DISK_ERR4
Predictive Failure Analysis Threshold Reached on Recovered Error	1 5D 00	1	1	SSA_DISK_ERR4
Logical Unit Not Ready Cause Not Reportable	2 04 00	6	2	SSA_DISK_ERR4
Logical unit is in the process of becoming ready	2 04 01	7	2	SSA_DISK_ERR4
Logical Unit Not Ready, initialization command required	2 04 02	6	2	SSA_DISK_ERR4
Logical Unit Not Ready, Format in Progress	2 04 04	4	2	SSA_DISK_ERR4
Medium Format Corrupted Reassign Failed	2 31 00	8	1	SSA_DISK_ERR4
Format Command Failed	2 31 01	4	1	SSA_DISK_ERR4
Diagnostic Failure	2 40 80	8	2	SSA_DISK_ERR4
Diagnostic Failure	2 40 85	14	1	SSA_DISK_ERR4
Diagnostic Failure	2 40 B0	9	1	DISK_ERR4
Logical Unit Failed Self-Configuration	2 4C 00	8	2	SSA_DISK_ERR4

Table 62. SCSI Sense Key/Code/Qualifier recovery procedures (continued)

Description	**SCSI K/C/Q**	**ERP#**	**Log**	**Template**
Write Error - Auto-Reallocation Failed	3 0C 02	5	1	DISK_ERR1
Write Error - Recommend Reassignment	3 0C 03	5	1	DISK_ERR1
Unrecovered Read Error	3 11 00	5	1	SSA_DISK_ERR2
Unrecovered Read Error - Auto Reallocation Failed	3 11 04	5	1	DISK_ERR1
Unrecovered Read Error - Recommend Reassignment	3 11 0B	5	1	DISK_ERR1
Recorded Entity Not Found	3 14 00	5	1	DISK_ERR1
Record Not Found	3 14 01	5	1	SSA_DISK_ERR2
Record Not Found - Recommend Reassignment	3 14 05	5	1	DISK_ERR1
Data Synchronization Mark Error	3 16 00	5	1	SSA_DISK_ERR2
Data Synchronization Mark Error - Recommend Reassignment	3 16 04	5	1	DISK_ERR1
Defect List Error in Primary List	3 19 02	4	1	SSA_DISK_ERR4
Defect List Error in Grown List	3 19 03	4	1	SSA_DISK_ERR4
Medium Format Corrupted Reassign Failed	3 31 00	4	1	SSA_DISK_ERR4
Format Failed	3 31 01	4	1	SSA_DISK_ERR4
Internal Target Failure	3 44 00	4	1	DISK_ERR1
No Index/Sector Signal	4 01 00	13	2	SSA_DISK_ERR4
No Seek Complete	4 02 00	13	2	SSA_DISK_ERR4
Peripheral Device Fault	4 03 00	13	2	SSA_DISK_ERR4
Track Following Error	4 09 00	13	2	SSA_DISK_ERR4
Unrecovered Read Error in Reserved Area	4 11 00	4	1	SSA_DISK_ERR4
Recorded Entity Not Found	4 14 00	13	2	SSA_DISK_ERR4
Record Not Found - Reserved Area	4 14 01	4	1	SSA_DISK_ERR4
Random Positioning Error	4 15 00	13	2	SSA_DISK_ERR4
Positioning Error Detected by Read of Medium	4 15 02	13	2	SSA_DISK_ERR4
Data Synchronization Mark Error in Reserved Area	4 16 00	4	1	SSA_DISK_ERR4
Defect List Error in Primary List	4 19 02	4	1	SSA_DISK_ERR4
Defect List Error in Grown List	4 19 03	4	1	SSA_DISK_ERR4
Medium Format Corrupted Reassign Failed	4 31 00	5	1	SSA_DISK_ERR4
No Defect Spare Location Available	4 32 00	4	1	SSA_DISK_ERR4
Defect list update failure	4 32 01	4	1	SSA_DISK_ERR4
Diagnostic Failure	4 40 80	8	2	SSA_DISK_ERR4
Diagnostic Failure	4 40 85	14	1	SSA_DISK_ERR4
Diagnostic Failure	4 40 90	8	2	SSA_DISK_ERR4
Diagnostic Failure	4 40 A0	8	2	SSA_DISK_ERR4
Diagnostic Failure	4 40 B0	9	1	DISK_ERR4

Table 62. SCSI Sense Key/Code/Qualifier recovery procedures (continued)

Description	SCSI K/C/Q	ERP#	Log	Template
Diagnostic Failure	4 40 C0	8	2	SSA_DISK_ERR4
Diagnostic Failure	4 40 D0	8	2	SSA_DISK_ERR4
Internal Target Failure	4 44 00	13	2	SSA_DISK_ERR4
Spindles Not Synchronized	4 5C 02	15	2	SSA_DISK_ERR4
Parameter List Length Error	5 1A 00	10	2	SSA_DISK_ERR1
Invalid Command Operation Code	5 20 00	10	2	SSA_DISK_ERR1
Logical Block Address out of Range	5 21 00	10	2	SSA_DISK_ERR1
Invalid Field in CDB	5 24 00	10	2	SSA_DISK_ERR1
Logical Unit Not Supported	5 25 00	10	2	SSA_DISK_ERR1
Invalid Field in Parameter List	5 26 00	10	2	SSA_DISK_ERR1
Not Ready To Ready Transition, (Medium may have changed)	6 28 00	9	0	-
Power On Reset, or Reset Message occurred	6 29 00	9	0	SSA_DISK_ERR1
Power On occurred	6 29 01	9	0	-
Total Reset received	6 29 02	9	0	-
Reset SMS received	6 29 03	9	0	-
Internal self-initiated reset occurred	6 29 04	9	0	-
Mode Parameters Changed	6 2A 01	9	0	-
Log Parameter Changed	6 2A 02	9	0	-
Commands Cleared by Another Initiator	6 2F 00	9	0	-
Microcode has been changed	6 3F 01	9	0	-
Spindles Synchronized	6 5C 01	13	0	-
Spindles Not Synchronized	6 5C 02	15	2	SSA_DISK_ERR4
Write Protected	7 27 00	4	2	SSA_DISK_ERR4
Internal Target Failure	B 44 00	13	2	SSA_DISK_ERR4
Overlapped Commands Attempted	B 4E 00	16	2	SSA_DISK_ERR1
Miscompare During Verify Operation	E 1D 00	4	1	SSA_DISK_ERR4
Invalid KCQ (Key = 1)	1 xx xx	1	1	SSA_DISK_ERR2
Invalid KCQ (Key = 2)	2 xx xx	6	2	SSA_DISK_ERR4
Invalid KCQ (Key =3)	3 xx xx	5	1	DISK_ERR1
Invalid KCQ (Key = 4)	4 xx xx	13	2	SSA_DISK_ERR4
Invalid KCQ (Key = 5)	5 xx xx	10	2	SSA_DISK_ERR1
Invalid KCQ (Key = 6)	6 xx xx	9	0	
Invalid KCQ (Key = 7)	7 xx xx	4	2	SSA_DISK_ERR4
Invalid KCQ (Key = B)	B xx xx	16	2	SSA_DISK_ERR4
Invalid KCQ (Key = E)	E xx xx	4	1	SSA_DISK_ERR4
Invalid KCQ (Any other Key)	x xx xx	4	1	SSA_DISK_ERR4

Appendix A. Identifier Values

This section supplies the values of the identifiers referenced elsewhere in this document. All values are decimal, except where shown as hexadecimal.

Registry Transactions

Name	Value
FN_REGY_SystemVersionInfo	10
FN_REGY_GatewayNodeList	11
FN_REGY_DriverGatewayNodeList	12
FN_REGY_ServiceList	13
FN_REGY_ConnectForNodeChange	14
FN_REGY_DiscForNodeChange	15
FN_REGY_NodeChangeToRegistry	16
FN_REGY_NodeChangeFromRegistry	17
FN_REGY_ConnectForErrorLogging	18
FN_REGY_DiscForErrorLogging	19
FN_REGY_LogErrorToRegistry	20
FN_REGY_LogErrorFromRegistry	21
FN_REGY_ConnectForResrcChange	22
FN_REGY_DiscForResrcChange	23
FN_REGY_ResrcChangeToRegistry	24
FN_REGY_ResrcChangeFromRegistry	25
FN_REGY_ResrcList	26
FN_REGY_GetTempResrcID	27
FN_REGY_ConnectForHealthCheck	28
FN_REGY_DiscForHealthCheck	29
FN_REGY_HealthCheckToRegistry	30
FN_REGY_HealthCheckFromRegistry	31
FN_REGY_SerialNumberSearch	32
FN_REGY_TestResrcsReady	33
FN_REGY_SetClusterNumber	34
FN_REGY_TestOneResrcReady	35
FN_REGY_SyncHCheckToRegy	36
FN_REGY_SyncHCheckFromRegy	37
EV_Rebooting	1
EV_NodeDead	2
EV_Rebooted	3
EV_NodeUnreliable	4
EV_NodeOpen	5
EV_NodeClose	6
RS_NotKnown	0
RS_Offline	1
RS_Online	2
CC_Add	01h
CC_SetOnline	02h
CC_SetOffline	04h
CC_Remove	08h

Name	Value
TY_Disk	0
TY_Adapter	1
TY_Enclosure	2
SD_Code	0
SD_CodeAsn	1
SD_CodeAsnCsn	2
SR_NoSynchro	0
SR_Synchro	1
LP_Unknown	0
LP_Loop	1
LP_String	2
LG_Unknown	0
LG_Legal	1
LG_Illegal	2
MN_Unknown	0
MN_Master	1
MN_NonMaster	2

ISAL Transactions

Name	Value
FN_ISALMgr_Inquiry	40
FN_ISALMgr_HardwareInquiry	41
FN_ISALMgr_SetOwningModuleType	42
FN_ISALMgr_AssignManualResrcID	43
FN_ISALMgr_GetPhysicalResrcIDs	44
FN_ISALMgr_TestResrcsReady	45
FN_ISALMgr_VPDInquiry	46
FN_ISALMgr_Characteristics	47
FN_ISALMgr_Statistics	48
FN_ISALMgr_FlashIndicator	49
FN_ISALMgr_Open	50
FN_ISAL_Close	51
FN_ISAL_Read	52
FN_ISAL_Write	53
FN_ISAL_Format	54
FN_ISAL_Progress	55
FN_ISAL_Lock	56
FN_ISAL_Unlock	57
FN_ISAL_Test	58
FN_ISAL_SCSI	59

Name	Value
FN_ISAL_Download	60
FN_ISALMgr_QueryFilterType	61
FN_ISAL_Fence	62
FN_ISALMgr_TestOneResrcReady	63
FN_ISALMgr_Get PhysSvcAndRIDs	64
FN_ISALMgr_ServiceMode	65
FN_ISALMgr_NetworkInquiry	66
FN_ISALMgr_Preferences	67
FN_ISAL_Flush	68
FN_ISALMgr_LockQuery	69
FN_ISAL_InitSurf	70
MD_ISAL	0
MD_SCSI	1
MD_Service	2
MD_ISAL_HA	3
AT_All	0
AT_ReadOnly	1
AT_WriteOnly	2
SM_DenyNone	0
SM_DenyAll	1
SM_DenyWrite	2
SM_DenyRead	3
SM_DenyNothing	4
FF_Verify	01h
FF_ExtendedFlags	02h
FF_NoCache	04h
FF_Split	08h
FF_ReadDisk	10h
FF_FastWrite	20h
FF_ISALReservedArea	40h
FF_ReassignWrite	80h
EF_NoDestage	01h
EF_RelocationArea	02h
EF_Idisk	04h
EF_IdiskSector	08h
EF_Override	10h
EF_NoIdisk	20h
EF_NoRetryOnError	40h
RF_ReassignWarn	01h
RF_DriveWarn	02h
RF_RewriteWarn	04h
RF_BlockWarn	08h
UL_Normal	0
UL_Forced	1
LT_Normal	0
LT_AdapID	1
ST_Good	0
ST_Failed	1
ST_LossRedundancy	2
ST_FormatInProgress	3

Name	Value
VP_NoEVPD	0
VP_EVPD	1
TT_Test	0
TT_Diag	1
AS_Warning	1
AS_Success	0
AE_UnknownFunction	-1
AE_Busy	-2
AE_Failure	-3
AE_HardwareError	−4
AE_ParityNotValid	−6
AE_MediaReadOnly	−7
AE_IllegalRequest	−100 to −50
AE_TableFull	−9
AE_InvalidRID	−10
AE_InvalidSignature	−11
AE_AccessDenied	−12
AE_NotReady	−13
AE_ReservationConflict	−14
AE_WriteProtect	−15
AE_NotInTable	−16
AE_Offline	−17
AE_MediumError	−18
AE_SCSIError	−20
AE_LogOpen	−21
AE_FencedOut	−22
AE_FormatDegraded	−23
AE_FormatInProgress	−24
AE_ClusterNumberNotKnown	−25
AE_SSAString	−26
AE_AvoidReadMe	−27
AE_AvoidWrite	−28
AE_NotLocal	−29
AE_NotYetImplemented	−30
AE_RetryWhenMemory	−31
AE_NotFound	−32
AE_FlushCompFailure	−33
AE_InServiceMode	−34
AE_OfflineTimeout	−35
AE_TcbAborted	−39
AE_NotSupported	−40
AE_NonIsal	−41
AE_OtherAdapterServiceMode	−42
AE_NonIdem	-43
AE_RequireLogical	-44
AE_RequirePhysical	-45

Name	Value
AE_BadSerialNumber	−500+0
AE_BadOldSerialNumber	−500+1
AE_BadNewSerialNumber	−500+2
AE_BadComponentCount	−500+3
AE_BadComponentSerialNumber	−500+4
AE_BadResrcSerialNumber	−500+5
AE_BadOldComponentSerialNumber	−500+6
AE_BadNewComponentSerialNumber	−500+7
AE_BadParameterValues	−500+8
AE_ArrayIsBroken	−500+9
AE_SetOMTFailed	−500+10
AE_BadExchangeCandidate	−500+11
AE_FiltersOnly	−500+12
AE_NotFilters	−500+13
AE_NvramError	−500+14
AE_InvalidCandidateRequest	−500+15
AE_InvalidCreateRequest	−500+16
AE_ReadOnlyParameterValue	−500+17
AE_ArrayIsBrokenOrDegraded	−500+18
AE_PhysWrapped	−500+19
AE_PhysCertifying	−500+20
AE_PhysFormatting	−500+21
AE_NotOpen	−500+22
AE_BadComponent	−500+23
AE_ConfirmRequired	−500+24
AE_InvalidMetaRequest	−500+25
AE_PhysIniting	−500+26
AE_DataLossWillOccur	−500+27
AE_NoHotSpareAvailable	−500+28
AE_CoupleOffline	−500+29
AE_CoupleCopying	−500+30
AE_CoupleDegraded	−500+31
AE_NvRamDefective	−200+0
AE_NotApplicableForResourceType	−200+1
AE_CantDeleteInUseResource	−200+2
OM_NotOwned	1
OM_DriverAdapters	2
OM_DriverPhysical	3
OM_DriverManualDisk	4
OM_DriverAutomaticDisk	5
OM_FastWriteFilter	E
OM_RAID0Filterr	F
OM_RAID5Filter	K
OM_Disowned	W
OM_NvramEntry	X
OM_HotSpareDisk	Y
OM_ListAll	0x9D

Name	Value
DL_NoSave	0
DL_Save	1
FF_Normal	0
FF_Force	1
FL_FenceOut	1
FL_FenceIn	2
FC_Add	1
FC_Remove	2
SI_NonIdem	0001
SI_Override	0002
SI_AutoSense	0100
HF_Unknown	0
HF_MotorFail	1
HI_NotImmediate	0
HI_Immediate	1
FM_Change	0
FM_CompareAndSwap	1
NI_NetworkA	0
NI_NetworkB	1
NI_Device	fc
NI_Unknown	fd
NI_DontCare	fe
NI_NullNetwork	ff
NI_Shared	NI_NetworkA
NI_Local	NI_NetworkB

Adapter Services

Name	Value
FN_ADAP_Control	79
FN_ADAP_TransferFromHost	80
FN_ADAP_TransferToHost	81
FN_ADAP_TargetTransfer	83
FN_ADAP_ConnectForHostTransfer	84
FN_ADAP_DiscForHostTransfer	85
FN_ADAP_GetClusterNumber	86
FN_ADAP_AdapterHealthCheck	90
FN_ADAP_ListSSANodes	91
FN_ADAP_QueryNodes	93
FN_ADAP_GetAdapterUID	94
FN_ADAP_SetTime	95
FN_ADAP_SetMasterPriority	96
FN_ADAP_GetMasterPriority	97
FN_ADAP_GetSupportLevel	98
FN_ADAP_QueryPort	99
FN_ADAP_ForceWrap	100
FN_ADAP_GetStatistics	101
NI_NetworkA	0
NI_NetworkB	1

Name	Value
PI_Port1	1
PI_Port2	2
TT_VSC	1
TT_DataTransfer	2
TT_CNUM	3
SC_Loop	0
SC_String	1
SC_IllegalString	2
DV_NoDevices	0
DV_Devices	1
QP_CLE	01
QP_CFC	02
AE_MissingCluster	-700+0
AE_RoutingError	-700+1
AE_RemoteTimeout	-700+2
AE_TransferFail	-700+3
AE_TargetNotAvailable	-700+4
AE_TargetReceiverFull	-700+5
AE_TargetTransferTooLarge	-700+6
AE_InConfig	-700+7
AE_QNTimedOut	-700+8

Service / Transaction Directives

Name	Value
MF_System	1
MF_Application	2
MF_Gateway	3
SF_Stop	1
SF_Ping	2
SF_Finger	3
SF_PowerFailure	4
SF_Version	5
OT_Parms	1
OT_Fetch	2
OT_Store	3
OT_Status	4
OT_Done	5
OT_FastDone	6

Name	Value
SN_Router	0
SN_Registry	1
SN_TimeServer	2
SN_ErrorLogger	3
SN_SSAGS	4
SN_SSADS	5
SN_CfgAgent	6
SN_AdapterService	7
SN_Proxy	8
SN_PeerRouter	9
SN_RemoteDiskHead	10
SN_RemoteDiskTail	11
SN_HAManager	12
SN_DiskService	16
SN_DebugService	17
TP_Unused	0
TP_Unknown	1
TP_Router	2
TP_ISAL	3
TP_FileSystem	4
TP_Database	5
TP_Resource	6
TP_Registry	7
TP_TimeServer	9
TP_ErrorLogger	10
TP_SSAGS	11
TP_SSADS	12
TP_Window	13
TP_BlowTorch	14
TP_CfgAgent	15
TP_AdapterService	16
TP_Debug	17
TP_Nvram	18
TP_Proxy	19
TP_PeerRouter	20

Node Numbers

Name	Value
NN_Local	0 The local node
NN_Host	1 The host
NN_RemoteHost	2 Remote Host
NN_AdapterA	11 First Adapter on bus 1
NN_AdapterAEnd	18 Last Adapter on bus 1
NN_AdapterB	21 First Adapter on bus 2
NN_AdapterBEnd	28 Last Adapter on bus 2
NN_DaughterA	31 First Daughter on bus 1
NN_DaughterAEnd	38 Last Daughter on bus 1
NN_DaughterB	41 First Daughter on bus 2
NN_DaughterBEnd	48 Last Daughter on bus 2
NN_UserStart	49 First User mode node
NN_UserEnd	63 Last User mode node
NN_PeerStart	64 First Peer-to-Peer node
NN_PeerEnd	127 Last Peer-to-Peer node
NN_NewAdapter	128
NN_NewAdapterEnd	191

Configuration / Array Identifiers

Name	Value
FN_IACL_Command	101
FN_IACL_Register	102
FN_IACL_Unregister	103
FT_NotOwned	A
FT_DriverAutomaticDisk	B
FT_DriverManualDisk	C
FT_PhysicalDisk	D
FT_FastWriteFilter	E
FT_RAID0Filter	F
FT_RAID5Filter	K
FT_Unsupported	U
FT_Disowned	W
FT_NVRAMEntry	X
FT_HotSpareDisk	Y
FT_BlankReserved	Z
FT_Adapter	Z+1

Name	Value
FC_IACLVersion	1
FC_ResrcCount	2
FC_ResrcList	3
FC_ResrcView	4
FC_CandidateCount	5
FC_CandidateList	6
FC_ResrcCreate	7
FC_ResrcDelete	8
FC_ResrcRename	9
FC_ComponentView	10
FC_ComponentExchange	11
FC_QueryMetaResrcParams	12
FC_ModifyResrcParams	13
FC_FlashIndicator	14
FC_VPDInquiry	15
FC_HardwareInquiry	16
FC_ComponentExchCandCount	17
FC_ComponentExchCandList	18
FC_AdapterVPD	19
FC_SyncHealth	20
FC_Wrap	21
FC_Unwrap	22
FC_UnwrapAll	23
FC_Test	24
FC_Format	25
FC_Certify	26
FC_Read	27
FC_Write	28
FC_AdapterSN	29
FC_CacheFormat	30
FC_InitSurf	31
FC_HotspareCfgStatus	32
FC_HotsparePoolList	33
FC_HotsparePoolView	34
FC_ReadArrayHotspareParams	35
FC_WriteArrayHotspareParams	36
FC_DeconfigureDisk	37
FC_CoupleArray	38
FC_UncoupleArray	39
FC_ReadUncoupleMetaData	40
FC_CoupleCompCandCount	41
FC_CoupleCompCandList	42
FC_CoupleResrcCandCount	43
FC_CoupleResrcCandList	44
FC_CoupleArrayComponentView	45
FC_WriteUncoupleMetaData	46
FS_CandOnline	20
FS_CandOffline	21

Name	Value
FS_ResrcOffline	40
FS_ResrcOnline	41
FS_ResrcOnlineNonDeg	42
FS_ResrcOnlineDeg	43
FS_ResrcOnlineRebuild	44
FS_ResrcOnlineExposed	45
FS_ResrcUnknown	46
FS_ResrcWringCache	47
FS_ResrcFormatting	50
FS_ResrcFormatFailed	51
FS_ResrcCertifying	52
FS_ResrcCertifyFailed	53
FS_CompNotPresent	60
FS_CompNotPresentDeconf	61
FS_CompNotPresentBlank	62
FS_CompPresent	65
FS_CompPresentRebuild	66
FS_CompPresentRebuildMe	67
FS_CompUnknown	68
FS_CompIllegalNetwork	69
FS_ResrcInUse	70 InUse (X%used)
FS_ResrcDormant	71 Dormant (X%used)
FS_ResrcWrapped	80 Wrapped, in service mode
FU_NewBat	1
HC_NoHotSpareProtection	1
HC_PoolShortConfig	2
HC_PoolLessMin	4
HC_PoolOutOfSync	8
HC_NotProtectingComp	10
HC_UnpreferredExchange	20
HT_HotSpare	0x01
HT_Array	0x02
HT_Component	0x04
HT_TooLarge	0x10
HT_WrongPool	0x20
HP_HotSpareEnabled	1
HP_HotSpareExact	2
HP_HotSparePreferred	4

Name	Value
SDS_BLOCKSIZE	1
SDS_DISK_NUMBER	2
SDS_STRIPE_SIZE	3
SDS_STRETCH_SIZE	4
SDS_MODE_FLAGS	5
SDS_STRIDE_SIZE	6
SDS_HOT_SPARE_ENABLED	7
SDS_HOT_SPARE_EXACT_SIZE	8
SDS_REBUILD_PRIORITY	9
SDS_SPEC_READ	10
SDS_DATA_SCRUB_ENABLED	11
SDS_DATA_SCRUB_RATE	12
SDS_SIZE	13
SDS_CACHE_SIZE	14
SDS_INITIALIZE	15
SDS_DELAY	16
SDS_SPLIT_RESOLUTION	17
SDS_RO_WHEN_EXPOSED	18
SDS_LAZY_PARITY_WRITE	19
SDS_PAGE_ALIGN_SPLIT	20
SDS_NETWORK_ID	21 Network ID
SDS_GEOM_SECT	22 Geometry hint
SDS_READ_CACHE_DISABLE	23 Read cache disabled
SDS_READ_AHEAD_ENABLE	24 Read ahead enabled
SDS_BAD_PTY_STRIDE	25
SDS_BAD_COMPONENT_STRIDE	26
SDS_BAD_STRIPE	27
SDS_MIN_LBA	28 Fastwrite min LBA of cached LBA range
SDS_MAX_LBA	29 Fastwrite max LBA of cached LBA range

Name	Value
SDS_MAX_WRITE_LENGTH	30 Fastwrite max length of writes cached
SDS_ALLOW_DELETE	31 Fastwrite allow data in cache to be deleted
SDS_MIRROR_ENABLE	32 Fastwrite snoop data into battery- backed SRAM
SDS_SKIP_WR_REBUILD	33 Skip Write Rebuild
SDS_SPLIT_CONFIRM	34 Split Confirm (RAID-1 only)
SDS_FAST_DECONFIGURE	35 Fast Deconfigure
SDS_CACHE_FSW	36 Set if full stride writes are to be cached
SDS_NO_INITIAL_REBUILD	39 No initial parity rebuild
SDS_NO_SHUTDOWN_WHEN_IDLE	41 Do not shutdown resource when idle
SDS_BYPASS_ONE_WAY	42 Bypass cache if one-way
SDS_POOL_SYNCHRONIZED	43 Hotspare pool is synchronized
SDS_HOTSPARE_POOLNUM	44 Hotspare pool number
SDS_HOTSPARES_IN_POOL	45 Number of hotspares in pool (actual)
SDS_HOTSPARES_CFG_IN_POOL	46 Hotspares in pool (configured)
SDS_HOTSPARES_MINIMUM	47 Minimum number of hotspares in pool
SDS_HOTSPARE_SPLITS	48 Hotspare splits
SDS_HOTSPARE_PREFERRED	49 Hotspare preferred
SDS_DEFINED_HOTSPARE_POOL_ONLY	50 Hotspare preferred pool only
SDS_UNCOPIED_STRIPS	51 Number of Uncopied strips
SDS_ARRAY_COUPLED	52 Array coupled to this array
SDS_COPY_RATE	53 Array copy rate
SDS_COPY_VERIFY_WRITES	54 Array copy will use verify writes
SDS_ENABLE_EARLY_RECONSTRUCT	55 Control if data is reconstructed on long ERPs
SDSF_MB	00000001h
SDSF_KB	00000002h
SDSF_PERCENT	00000004h
SDSF_ON_OFF	00000008h
SDSF_BYTES	00000010h
SDSF_SECONDS	00000020h
SDSF_MINUTES	00000040h
SDSF_HOURS	00000080h
SDFS_LBA	00000100h
SDFS_BLOCKS	00000400h
SDFS_STRIDES	00000800h
SDSF_READONLY	80000000h
SDSF_UNIQUE	40000000h

Appendix B. Notices

References in this book to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Subject to IBM’s valid intellectual property or other legally protectable rights, any functionally equivalent product, program, or service may be used instead of the IBM product, program, or service. The evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, are the responsibility of the user.

Any examples of parameters or definitions are for guidance only. Some details may differ from the requirements in your environment. Contact your IBM representative if you need further assistance.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license enquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, New York 10594, U.S.A.

Trademarks

The following terms are trademarks of International Business Machines Corporation in the United States, or other countries, or both:

IBM	@server
Micro Channel	

Other company, product, and service names may be trademarks or service marks of others.

Index

Printed in the U.S.A.

SA33-3286-02

Spine information:

IBM Advanced SerialRAID Adapters Technical Reference