To: T10 Technical Committee
From: Rob Elliott, Compaq Computer Corporation (Robert.Elliott@compaq.com)
Date: 23 February 2002
Subject: T10/02-071r0 SAM-2 Asynchronous event reporting RPC

Revision History

Revision 0 (23 February 2002) first revision

Related Documents

sam2r21 - SCSI Architecture Model - 2 revision 21 (Ralph Weber)

Overview

SAM-2 does not include a remote procedure call (RPC) and protocol services model for asynchronous event reporting (AER).

AER does not map well into the remote procedure call request/response model used for Execute Command() and task management functions, since it's really a target-initiated activity. The only "request" made by the application client is enabling AER by writing the control mode page; for this it receives many "responses" from the device server (the asynchronous events).

If the device server is allowed to be the client, however, it can be modeled. The server is an application client. Whenever an asynchronous event occurs, the device server invokes an RPC to report it to the application client.

Application client is not a perfect fit to receive, since it's traditionally a transient entity. However, there is currently no "application manager" to parallel the "task manager" in a target device that always exists. It may be better to add that to the object model and use it; this proposal does not do so.

The protocol services to support the RPC can also be constructed to describe how the report is delivered. They are similar to the data transfer protocol services, which are also initiated by the target. A four step process is described.

Suggested Changes

3.2 Acronyms

ACA Auto Contingent Allegiance (see 3.1.5)
AER Asynchronous Event Reporting
…

4.1 Introduction

…

Requirements that apply to each SCSI protocol standard are specified in the SCSI protocol service model described in 5.4, 5.8.4.2.3, and 6.9. The model describes required behavior in terms of layers, objects within layers and protocol service transactions between layers.

4.3 The SCSI client-server model

As shown in figure 6, each SCSI target device provides device services performed by the logical units under the control of the target and task management functions performed by the task manager. A logical unit is an object that implements one of the device functional models described in the SCSI command standards and processes SCSI commands such as reading from or writing to the media. Each pending SCSI command or series of linked commands defines a unit of work to be performed by the logical unit. Each unit of work is represented within the target by a task that may be externally referenced and controlled through requests issued to the task manager.

All command and task management requests originate from application clients residing within a SCSI initiator device. An application client represents a thread of processing whose functionality is independent of the interconnect and SCSI protocol. In an implementation, that thread could correspond to the device driver and any other code within the operating system that is capable of managing I/O requests without requiring knowledge of the interconnect or SCSI protocol. In the architecture model, an application client is created to issue a single SCSI command or task management function. An application client ceases to exist once the command or task management function ends. Consequently, there is one application client for each pending command or task management request. Within the initiator, one or more controlling entities, whose definition is outside the scope of the architecture model, oversee the creation of and interaction among application clients.

Asynchronous event reporting requests originate from device servers residing within a SCSI target device. A SCSI initiator device creates application clients to service these requests.

As described in 4.2, each request takes the form of a procedure call with arguments and a status to be returned.
[insert paragraph here, to make "… may request…" lead the paragraph.]
An application client may request processing of a SCSI command through an Execute Command request directed to the device server within a logical unit (see 5.1). Each ~~device service~~ request contains a CDB, defining the operation to be performed, along with a list of command specific inputs and other parameters specifying how the command is to be processed. If supported by a logical unit, a sequence of linked commands may be used to define an extended I/O operation.

A task is an object within the logical unit representing the work associated with a command or series of linked commands. A new command or the first in a series of linked commands causes the creation of a task. The task persists until a command completion response is sent or until the task is ended by a task management function or exception condition. For an example of the processing for a single command see 5.7.1. For an example of linked command processing see 5.7.2.

An application client may request processing of a task management function through a task management function request directed to the task manager within the logical unit (see 6.1). The interactions between the task manager and application client when a task management request is processed are shown in 6.10.

A device server may request reporting of an asynchronous event through a Report Asynchronous Event request directed to an application client in a SCSI initiator device (see 5.8.4.2.2).

…

5.8.4.2 Asynchronous Event Reporting

5.8.4.2.1 Asynchronous Event Reporting overview

Asynchronous Event Reporting is used by a logical unit to signal another device that an asynchronous event has occurred. The mechanism automatically returns sense data associated with the event. Each SCSI protocol standard shall describe a mechanism for Asynchronous Event Reporting. (In this subclause, references to Asynchronous Event Reporting assume that the device to be notified has enabled asynchronous event reports from the target.) Support for asynchronous event reporting is a logical unit option.

NOTE 11 - A SCSI device that is capable of producing asynchronous event reports at initialization time should provide means to defeat these reports. This may be done with a switch or jumper wire. Devices that implement saved parameters may alternatively save the asynchronous event reporting permissions either on a per SCSI device basis or as a system wide option.

Parameters managing the use of asynchronous event reporting are contained in the Control mode page (see SPC-2).

Asynchronous Event Reporting is used to signal a device that one of the four events listed below has occurred:
a) An exception condition was encountered after command completion;

b) A newly initialized device is available;
c) Some other type of unit attention condition has occurred; or
d) An asynchronous event has occurred.

An example of a) occurs in a device that implements a write cache. If the target is unable to write cached data to the medium, it may use an asynchronous event report to inform the initiator of the failure.

An example of b) is a logical unit that generates an asynchronous event report, following a power-on cycle, to notify other SCSI devices that it is ready to accept I/O commands.

An example of c) occurs in a device that supports removable media. Asynchronous event reporting may be used to inform an initiator of a not-ready-to-ready transition (medium changed) or of an operator initiated event (e.g., activating a write protect switch or activating a start or stop switch).

An example of d) is a sequential-access device performing a REWIND command with the IMMEDIATE bit set to one (see SSC-2). An asynchronous event report may be used to inform an initiator that the beginning of medium has been reached. Completion of a CD-ROM AUDIO PLAY command (see MMC-~~2~~3) started in the immediate mode is another example of this case. Sense data accompanying the report identifies the condition (see 5.8.4.1).

An exception condition encountered after command completion shall be reported to a specific initiator once per occurrence of the event causing it. The logical unit may choose to use an asynchronous event report or to return CHECK CONDITION status on a subsequent command, but not both. Notification of an exception condition encountered after command completion shall be reported only to the initiator or initiators that sent the affected task or tasks.

Asynchronous event reports may be used to notify devices that a system resource has become available. If a logical unit uses this method of reporting, the sense key in the AER sense data shall be set to UNIT ATTENTION.

5.8.4.2.2 Report asynchronous event remote procedure call

The device server uses the report asynchronous event remote procedure call to deliver an asynchronous event report to an initiator device whose application clients have previously enabled AER in the control mode page.

In some protocols (e.g. SPI-4, FCP-2), asynchronous event reporting is only supported when the application client and device server are each located in target/initiator devices. To send the asynchronous event report, the device server employs an application client and uses the target/initiator port in the initiator role. To receive the asynchronous event report, the application client employs a device server in its target/initiator device and uses the target/initiator port in its target role.

In other protocols (e.g., SRP, iSCSI), the asynchronous event report is delivered directly by the device server in a target device to an application client in an initiator device.

Request:

Service Response = Report Asynchronous Event (IN (I_T_L Nexus, Sense Data))

Input arguments:

I_T_L Nexus: An I_T_L nexus (see 4.10) indicating the source of the asynchronous event.
Sense Data: Sense data associated with the asynchronous event (see 5.8.4.3).

Service Response assumes one of the following values:

ASYNCHRONOUS EVENT REPORTED: An asynchronous event has been reported.

SERVICE DELIVERY OR TARGET FAILURE: The delivery of an asynchronous event report failed due to a service delivery failure or SCSI initiator device malfunction.

5.8.4.2.3 Asynchronous event reporting protocol services

5.8.4.2.3.1 Asynchronous event reporting protocol services overview

The asynchronous event reporting protocol services are used to process the Report Asynchronous Event remote procedure call as follows:

1) Device server: Send Asynchronous Event Report ();
2) Application client: Asynchronous Event Report Received();
3) Application client: Asynchronous Event Report Accepted(); and
4) Device server: Asynchronous Event Report Delivered().

5.8.4.2.3.2 Asynchronous event reporting indication

The device server uses a protocol service request to deliver asynchronous event reports to an application client.

Request (sent by device server to application client):

Send Asynchronous Event Report (IN (I_T_L Nexus, Sense Data))

Argument descriptions:

I_T_L Nexus: An I_T_L nexus (see 4.10) indicating the source of the asynchronous event.

Sense Data: Sense data associated with the event (see 5.8.4.3) .

5.8.4.2.3.3 Asynchronous event reporting indication

The application client receives a protocol service indication when an asynchronous event report arrives.

Indication (received by application client from device server):

Asynchronous Event Report Received (OUT (I_T_L Nexus, Sense Data))

Argument descriptions:

I_T_L Nexus: An I_T_L nexus (see 4.10) indicating the source of the asynchronous event.

Sense Data: Sense data associated with the event (see 5.8.4.3) .

5.8.4.2.3.4 Asynchronous event reporting response

The application client acknowledges an asynchronous event report with a protocol service response.

Response (sent by application client to device server):

Asynchronous Event Report Accepted (IN (I_T_L Nexus, Service Response))

Argument descriptions:

I_T_L Nexus: An I_T_L nexus (see 4.10) indicating the source of the asynchronous event.

Service Response: Service response (see 5.8.4.2.2).

5.8.4.2.3.5 Asynchronous event reporting confirmation

The device server receives protocol service confirmation that the asynchronous event report has been delivered.

Confirmation (received by device server from application client):

Asynchronous Event Report Delivered (IN (I_T_L_x Nexus), Service Response))

Argument descriptions:

I_T_L Nexus: An I_T_L nexus (see 4.10) indicating the source of the asynchronous event.
Service Response: Service response (see 5.8.4.2.2).

Implementation in different protocol standards [not proposed text for SAM-2]

SRP:

Send AER() = sending SRP_AER_REQ IU
AER Received() = receiving SRP_AER_REQ IU
AER Accepted() = sending SRP_AER_RSP IU
AER Delivered() = receiving SRP_AER_RSP IU

SPI-4/FCP-2:

Send AER = associated application client sending COMMAND IU through the target/initiator port acting as an initiator port, with a SEND CDB carrying sense data
AER Received() = associated device server receiving COMMAND IU through the target/initiator port acting as a target port
AER Accepted() = sending RESPONSE IU
AER Delivered() = receiving RESPONSE IU