Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303-4900

March 13, 1998 T10/98-124

To: NCITS Technical Committee T10

From: Bob Snively

Subject: SPC-2, Persistent Reservation: Additional proposed corrections

During implementation of various persistent reservation functions, the SPC standard has been found to be incomplete or unclear in a number of places. SPC-2 is the ideal place to correct these problems. In addition, the architectural study required to implement persistent reservation has revealed that some additional functions are required. This document proposes the necessary changes to SPC-2 and additionally proposes changes to SBC. My thanks to the many people who have contributed to this study, including Gerry Houlder, Andy Hisgen, and Brian Edgar.

1 Unregister capability

The service action description for the Register service action (7.13.1.1) needs to be clarified to clearly indicate that the unregister function is performed by performing a Register service action with the Service Action Reservation Key set to a zero value. At present, it is difficult to determine whether the reservation registration key resources are released by that action or not. This requires a change to 7.3.1.1 as shown below. Similar wording is required in other sections which are also indicated in this document.

7.3.1.1 Register

The PERSISTENT RESERVE OUT command executing a Register service action registers a reservation key with a device server without generating a reservation. For each initiator that performs a PERSISTENT RESERVE OUT Register service action, the device server shall retain the reservation key until the key is changed by a new PERSISTENT RESERVE OUT command with the Register service action from the same initiator ~~or until the key is reset to the default value of zero~~ or until the initiator registration is removed by one of the following actions:

a) powering down the logical unit, if the last APTPL received by the device server was zero (see 7.13.2);

b) performing a Clear service action;

c) performing a Preempt service action~~; or~~

d) performing a Preempt and Clear service action; or

e) performing a Register service action with the value of the Service Action Reservation key set to zero.

When the reservation key has been removed, ~~the default~~ a reservation key of zero shall be used when the initiator registers a reservation key again. When the reservation key has been removed, no information is reported for the initiator in the Report Keys service action and all persistent reservations associated with that initiator and reservation key shall be released.

2 Clarify CDB field requirements

The scope field and the type field are not defined and have no meaning for the Register and for the Clear service actions. Text in each of these service action descriptions must be installed to indicate that these fields should be set to zero by an application client, but shall be ignored by the device server whether zero or not.

In the last paragraph of section 7.13.1.1, describing the Register service action, the following change should be made.

The Register service action may be performed regardless of any active persistent reservations. All existing persistent reservations for the initiator receive the new reservation key. The Scope field and the Type field shall be ignored by the device server.

After the third paragraph of section 7.13.1.4, describing the Clear service action, the following change should be made.

The Scope field and the Type field for a Clear service action shall be ignored by the device server.

3 Proposed improvements to Release service action

The following text is proposed to clarify what actually is performed when a Release service action is performed when no such reservation exists. The text is modified in section 7.13.1.3, paragraph 1. In the first paragraph, the text is modified to clarify that there is no release performed when there is no reservation to be released. In the second paragraph, the modifications clarify the requirements for checking that a release service action is performed against a properly matching reservation. Item a) in the third paragraph is modified to indicate that power off actually removes reservation keys and releases any resources required to preserve them.

The PERSISTENT RESERVE OUT command performing a Release service action removes an active persistent reservation held by the same initiator. The fields associated with the Release service action shall match fields of the active persistent reservation. It shall not be an error to send a PERSISTENT RESERVE OUT specifying a Release service action when no persistent reservation exists from that initiator. In that case, no persistent reservation will

■

be released. The reservation key shall not be changed by the Release service action.

The device server shall return a CHECK CONDITION status for a PERSISTENT RESERVE OUT command that specifies the release of a persistent reservation matching the requesting initiator's ID and some but not all of the scope, reservation key, reservation type, and extent values. The sense key shall be set to ILLEGAL REQUEST and additional sense data shall be set to INVALID RELEASE OF ACTIVE PERSISTENT RESERVATION. Attempts to release persistent reservations where none of the scope, reservation key, reservation type, and extent values match an existing persistent reservation held by the requesting initiator shall not be considered errors.

An active persistent reservation may also be released by either of the following mechanisms:

a) Power off. When the most recent APTPL value received by the device server is zero (see 7.13.2), a power off performs a hard reset, clears all persistent reservations, and removes all registered reservation keys~~, setting them to the default value of zero~~ (see 7.13.1.1); or

b) Execution of a PERSISTENT RESERVE OUT command from another initiator with a Persistent Reserve service action of Preempt or Preempt and Clear.

The remaining paragraphs of 7.13.1.3 are unchanged.

4 Proposed clarification of Clear service action

The text of 7.13.1.4 needs to be clarified to indicate that the removal of a reservation key does not simply reset the value, but actually removes the reservation key and any resources related to the key.

The PERSISTENT RESERVE OUT command that successfully performs a Clear service action shall remove all persistent reservations for all initiators. All reservation keys shall be removed ~~and reset to the default value of zero~~ (see 7.13.1.1). Any commands from any initiator that have been accepted by the device server as nonconflicting shall continue normal execution.

5 Clarification of details in the Preempt service action

Paragraph 1 of section 7.13.1.5 needs to explicitly indicate that the preempting of a persistent reservation creates a reservation by the initiator performing the preempt service action.

The PERSISTENT RESERVE OUT command that successfully performs a Preempt service action shall remove all persistent reservations for all initiators that are registered with the Service Action Reservation key specified in the PERSISTENT RESERVE OUT parameter list. It shall also establish a per-

sistent reservation for the preempting initiator. Any commands from any initiator that have been accepted by the device server as nonconflicting shall continue normal execution. It shall not be an error to send a PERSISTENT RESERVE OUT specifying a Preempt service action when no persistent reservation exists for the initiator identified by the Service Action Reservation key.

The fourth paragraph of section 7.13.1.5 needs to be clarified to show that the removal of a reservation key does not simply reset the value, but actually removes the reservation key and any resources related to the key. In addition, this paragraph should indicate that execution of the Preempt service action with a Service Action Reservation Key equal to its own reservation key shall establish the new specified reservation. Other initiators having the same reservation key shall have the their reservation key removed.

The registration key for ~~the~~ other initiators that have been preempted shall be removed ~~and reset to the default value of zero~~ (see 7.13.1.1) by the Preempt service action. The reservation key for an initiator that has performed a Preempt service action with its own reservation key specified in the Service Action Reservation Key shall remain unchanged, although all other specified releasing actions and reservation actions shall be performed.

6 Clarification of details in the Preempt and Clear service action

Paragraph 1 of section 7.13.1.6 needs to explicitly indicate that the preempting of a persistent reservation creates a reservation by the initiator performing the Preempt and Clear service action.

The PERSISTENT RESERVE OUT command performing a Preempt and Clear service action removes all persistent reservations for all initiators that are registered with the Service Action Reservation key specified by the PERSISTENT RESERVE OUT parameter list. It also establishes a persistent reservation for the preempting initiator. Every command from the initiators being preempted shall be terminated as if an ABORT TASK task management function had been performed by the preempted initiator. It shall not be an error to send a PERSISTENT RESERVE OUT specifying a Preempt and Clear service action when no persistent reservation exists for the initiator identified by the Service Action Reservation key. However, if the key is registered, the Clear portion of the action shall execute normally.

The sixth paragraph of section 7.13.1.5 needs to be clarified to show that the removal of a reservation key does not simply reset the value, but actually removes the reservation key and any resources related to the key. In addition, this paragraph should indicate that execution of the Preempt and Clear service action with a Service Action Reservation Key equal to its own reservation key shall clear any commands for that device and establish the new

■

specified reservation. Other initiators having the same reservation key shall have the their reservation key removed.

The reservation key registered for ~~the~~ other initiators that have been preempted shall be removed ~~and reset to the default value of zero~~ (see 7.13.1.1) by the Preempt and Clear service action. The reservation key for an initiator that has performed a Preempt and Clear service action with its own reservation key specified in the Service Action Reservation Key shall remain unchanged, although all other specified clearing actions, releasing actions, and reservation actions shall be performed.

7 Correction of reservation key removal text in APTPL description

The sixth paragraph of section 7.13.2, needs to indicate that, if APTPL is zero, the reservation keys and related resources are actually removed by a power off, not merely set to the default value.

If the last valid APTPL bit value received by the device server is zero, the loss of power in the target shall release all persistent reservations and remove all reservation keys~~, setting them to the default value of zero~~ (see 7.13.1.1). If the last valid APTPL bit value received by the device server is one, the logical unit shall retain all persistent reservations and all reservation keys for all initiators even if power is lost and later returned. The most recently received valid APTPL value from any initiator shall govern logical unit's behavior in the event of power loss.

8 Exhaustion of reservation resources

At present, the SPC-2 document does not specify what action shall be taken if an attempt is made to create a reservation, but all reservation resources (memory space, allocated disk space, or other resources) are already used. The following addition after the last paragraph of section 7.13 is proposed.

If a PERSISTENT RESERVE OUT command is attempted, but there are insufficient device server resources to complete the operation, the device server shall return a CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and the additional sense data shall be set to DEFECT LIST UPDATE FAILURE.

9 Clarify actual exceptions to RESERVATION CONFLICT

The text of 7.13 is a little too restrictive in indicating what operations may proceed without encountering a RESERVATION CONFLICT. In fact, there are reservations that may cause RESERVATION CONFLICT even with Preempt or Preempt and Clear service actions. As an example, a Preempt operation that creates a new but conflicting reservation for the target performing the command will fail with RESERVATION CONFLICT status. In addition,

■

certain commands, including INQUIRY and REQUEST SENSE, will not conflict with any reservation. For this reason, it is proposed that the second paragraph after Table 44 in section 7.13 be modified as follows:

Commands from any initiator that conflict with a successfully established persistent reservation shall be rejected with a status of RESERVATION CONFLICT with the following exceptions:

a) PERSISTENT RESERVE IN shall not conflict with any persistent reservation;

b) PERSISTENT RESERVE OUT with a service action of Preempt shall not conflict with any persistent reservation; and

c) PERSISTENT RESERVE OUT with a service action of Preempt and Clear shall not conflict with any persistent reservation.

b) PERSISTENT RESERVE OUT shall not conflict with any persistent reservation except as discussed in the descriptions of the CDB fields, parameter list fields, and service actions of this command; and

c) Any command whose description states that the command shall not conflict with any persistent reservation shall not conflict with any persistent reservation.

10 Clarification of requirements on parameters

The paragraph just above table 47 in section 7.13.2 should be clarified to indicate that those parameters that are ignored shall not be checked by the device server. The ignored parameters may have any value.

Table 46 summarizes which fields are set by the application client and interpreted by the device server for each Service action and Scope value. Two PERSISTENT RESERVE OUT parameters are not summarized in table 46; Reservation key and APTPL, since they are specified above. A parameter that is ignored shall not be examined by the device server and may be set to any value by the application client.

11 Clarification of persistent reservation management method

The capability of devices using persistent reservation to create various kinds of shared reservations conditioned by the registry of a reservation key is a significant extension of the reservation model and should be included in section 5.3.2. The following text is proposed:

The Persistent Reservations management method is used among multiple initiators that require operations to be protected across initiator failures, which usually involve hard resets. Persistent reservations persist across recovery actions, to provide initiators with more detailed control over reservations recovery. Persistent reservations for failing initiators may be preempted by another initiator as part of the recovery process. Persistent reservations are retained by the device server until released, preempted, or until cleared by

■

mechanisms specified in this standard. Persistent reservations are optionally retained when power to the target is lost.

In addition to the reservations created directly by an initiator, the Persistent Reservations management method allows the creation of reservations which become automatically shared among those initiators which have registered with the device server.

12 Clarification of reservation conflicts for SPC-2 commands.

Several SPC-2 defined commands consider the case of logical unit reservations without considering whether the logical unit reservations are qualified by various further restrictions allowed by persistent reservations. In particular, persistent reservations may allow a reservation to be shared among all registered initiators and may allow read exclusive or write exclusive reservations. Some of the SPC-2 commands require further description when considering such reservations, as described below:

7.2 COMPARE command, second paragraph:

If reservations are active, they shall affect the execution of the COMPARE command as follows. A reservation conflict shall occur when a COMPARE command is received from an initiator other than the one holding or sharing through registration a logical unit reservation. The COMPARE command shall be evaluated for extent reservation conflicts as if the copy master were performing normal read operations even when an SCSI device is requested to compare with itself. For example, if a COMPARE is issued to logical unit 0 that requests the SCSI device to compare between data from logical unit 0 to data from logical unit 1, access to logical unit 1 also shall be evaluated for a reservation conflict. COMPARE commands shall be terminated with CHECK CONDITION status and the sense key shall be set to DATA PROTECT if any part of the compare operation is prohibited by an extent reservation.

The same change shall also be made for the following commands:

COPY, COPY AND VERIFY,

7.6, LOG SELECT command, second paragraph:

If reservations are active, they shall affect the execution of the LOG SELECT command as follows. A reservation conflict shall occur when a LOG SELECT command is received from an initiator other than the one holding or sharing through registration any ~~a~~ logical unit reservation. [Remaining text unchanged].

The similar changes shall also be made for the following commands:

■

LOG SENSE, MODE SELECT(6), MODE SENSE(6)

RECEIVE DIAGNOSTIC RESULTS, SEND DIAGNOSTIC

TEST UNIT READY

RECEIVE, SEND

7.14, PREVENT ALLOW MEDIUM REMOVAL command, second paragraph.

If reservations are active, they shall affect the execution of the PREVENT ALLOW MEDIUM REMOVAL command as follows. Receipt of a PREVENT ALLOW MEDIUM REMOVAL command with a Prevent value of zero shall not cause a reservation conflict under any circumstances. A reservation conflict shall occur when a PREVENT ALLOW MEDIUM REMOVAL command with a non-zero Prevent value is received from an initiator other than the one holding or sharing through registration any ~~a~~ reservation.

7.15, READ BUFFER command, second paragraph.

If reservations are active, they shall affect the execution of the READ BUFFER command as follows. A reservation conflict shall occur when a READ BUFFER command is received from an initiator other than the one holding or sharing through registration a logical unit reservation that allows a read operation. The READ BUFFER command shall not be affected by extent or element reservations.

7.25, WRITE BUFFER command, second paragraph.

If reservations are active, they shall affect the execution of the WRITE BUFFER command as follows. A reservation conflict shall occur when a WRITE BUFFER command is received from an initiator other than the one holding or sharing through registration a logical unit, extent, or element reservation that allows a write operation.

The following commands do not require any changes in the descriptive text for reservations:

CHANGE DEFINITION, INQUIRY

PERSISTENT RESERVE IN, PERSISTENT RESERVE OUT

RELEASE(10), RELEASE(6), REPORT LUNS, REQUEST SENSE

RESERVE(10), RESERVE(6),

■

13 Clarification of reservation conflicts for SBC commands

Changes similar to those required in SPC-2 are also required in SBC. However, SBC has now passed letter ballot and is not available for update. The general wording should indicate that reservations apply to logical units and extents and that reservations may either be explicit by the reserving initiator or implicit by sharing through registration. In addition, those commands that are considered as reads and those commands that are considered as writes should be clarified. The reservations are considered separately for each command in SPC-2. This editorial convention is considerably clearer than the collective description in section 5.1.8 in SBC and should be used for the next revision of SBC, since it allows the detailed discussion required by some of the commands. An example is SEEK, which is neither a read, nor a write, but could be related to either and is presently not well specified.

In addition to those general considerations, there are some errors in SBC that should be corrected.

SBC, section 5.1.8,

Paragraph 6, describing CHANGE DEFINITION is incorrect, defined in SPC-2 and should be deleted.

Paragraph 7, describing COMPARE, COPY, and COPY AND VERIFY is incorrect, defined in SPC-2, and should be deleted.

Paragraph 8 is incomplete in describing FORMAT UNIT, REZERO UNIT, and START STOP UNIT, and is incorrect in describing PREVENT ALLOW MEDIUM REMOVAL which is described in SPC-2.

Paragraph 9, while correctly describing INQUIRY and REQUEST SENSE, is already defined in SPC-2 and should be deleted.

Paragraph 10, describing LOG SELECT/SENSE, MODE SENSE, TEST UNIT READY, READ CAPACITY, READ BUFFER, WRITE BUFFER, and READ DEFECT DATA, is incomplete and conflicts with some of the SPC-2 definitions and should be deleted.

Paragraph 11, describing SEEK, LOCK UNLOCK, CACHE, PRE-FETCH, and SYNCHRONIZE CACHE, is incomplete and does not specify whether certain of these should be treated as reads or writes.

Paragraph 12, describing MODE SELECT, conflicts with some of the SPC-2 definitions and should be deleted.

Paragraph 13, describing SEND DIAGNOSTIC and RECEIVE DIAGNOSTIC RESULTS, is incomplete, conflicts with some of the SPC-2 definitions, and should be deleted.

The remaining paragraphs have incomplete descriptions.

■