Date: Nov 4, 1998

To: T10 Committee (SCSI)

From: George Penokie (IBM)

Subject: Persistent Reservations

5.3 Reservations

Reservations may be used to allow a device server to execute commands from a selected set of initiators. The device server rejects any commands from initiators outside the selected set of initiators by uniquely identifying initiators using protocol specific mechanisms.

Application clients may add or remove initiators from the selected set using reservation commands. If the application clients do not cooperate in the reservation protocol, data may be unexpectedly modified and deadlock conditions may occur.

The general description of reservations involves two groups of considerations;

a) the type of reservation established, and
b) the method used to establish, rescind, and manage the reservation.

There are limits on the combinations of reservation types available under some reservation management methods. See the reservations management commands descriptions for details.

The scope of a reservation shall be one of the following:

a) **logical unit reservations** - a logical unit reservation restricts access to the entire logical unit; or
b) **element reservations** - an element reservation restricts access to a specified element within a medium changer.

Reservations may be further qualified by restrictions on types of access (e.g., read, write, control). However, any restrictions based on the type of reservation are independent of the scope of the reservation. In addition, some methods of reservation management permit establishing reservations on behalf of another device in the same SCSI domain (third-party reservations).

The methods of managing reservations are identified by the commands associated with the methods. The methods of managing reservations are:

a) Reserve/Release - associated with the RESERVE(10), RELEASE(10), RESERVE(6), and RELEASE(6) commands (see 7.21, 7.17, 7.22, and 7.18, respectively); and
b) Persistent Reservations - associated with the PERSISTENT RESERVE OUT and PERSISTENT RESERVE IN commands (see 7.13 and 7.12, respectively).

Reservation restrictions are placed on commands as a result of access qualifiers associated with the type of reservation. The details of which commands are allowed under what types of reservations are described in table 1. For the reservation restrictions placed on commands for the Reserve/Release management method see table 1 column A. For the reservation restrictions placed on commands for the Persistent Reservations management method, see table 1 columns under B.

<u>If any element is reserved within a logical unit, that logical unit shall be considered reserved for the commands listed in table 1 and the accept/conflict information in the table shall apply.</u>

In table 1 the following key words are used:

allowed: Commands issued by initiators not holding the reservation or by initiators not registered when a registrants only persistent reservation is present should complete normally.

conflict: Commands issued by initiators not holding the reservation or by initiators not registered when a registrants only persistent reservation is present shall not be performed and the device server shall terminate the command with a RESERVATION CONFLICT status.

Commands from initiators holding a reservation should complete normally. The behavior of commands from registered initiators when a registrants only persistent reservation is present is specified in table 1 and table 2.

~~A command that does not explicitly read or write the medium may be checked for reservation conflicts at any time between the time the command is delivered to the device server and the time the task representing the command becomes the current task. A command that explicitly reads or writes the medium may be checked for reservation conflicts at any time between the time the command is delivered to the device server and the time the device server begins to modify the data on the medium.~~

A command that does not explicitly write the medium shall be checked for reservation conflicts before the command enters the current task state for the first time. ~~The command may be subject to subsequent reservations checks, and as a result of that reservation check, the command may be terminated with a RESERVATION CONFLICT.~~ Once the command has entered the current task state, it shall not be terminated with a RESERVATION CONFLICT due to a subsequent reservation.

A command that explicitly writes the medium shall be checked for reservation conflicts before the device server modifies the medium or cache as a result of the command. Once the command has modified the medium, it shall not be terminated with a RESERVATION CONFLICT due to a subsequent reservation.

For each command, this standard or a related command standard (see 3.1.11) defines the conditions that result in RESERVATION CONFLICT. Depending on the particular command standard the conditions are defined in that standard's device model clause or in the clauses that define the specific commands. Annex A contains the RESERVATION CONFLICT information for some of the command sets.

Table 1 - SPC commands that are allowed in the presence of various reservations

Command	Addressed LU is reserved by another initiator (A)	Addressed LU has this type of persistent reservation held by another initiator (B)				
		From any initiator		From Registered Initiator (All RO types)	From Initiator Not Registered	
		Write Excl	Excl Access		Write Excl RO	Excl Access RO
COMPARE	conflict	allowed	conflict	allowed	allowed	conflict
COPY	conflict	conflict	conflict	allowed	conflict	conflict
COPY & VERIFY	conflict	conflict	conflict	allowed	conflict	conflict
INQUIRY	allowed	allowed	allowed	allowed	allowed	allowed
LOG SELECT	conflict	conflict	conflict	allowed	conflict	conflict
LOG SENSE	allowed	allowed	allowed	allowed	allowed	allowed
MODE SELECT(6)/MODE SELECT(10)	conflict	conflict	conflict	allowed	conflict	conflict
MODE SENSE(6)/MODE SENSE(10)	conflict	conflict	conflict	allowed	conflict	conflict
PERSISTENT RESERVATION IN	conflict	allowed	allowed	allowed	allowed	allowed
PREVENT/ALLOW (prevent=0)	allowed	allowed	allowed	allowed	allowed	allowed
PREVENT/ALLOW (prevent<>0)	conflict	conflict	conflict	allowed	conflict	conflict
READ BUFFER	conflict	conflict	conflict	allowed	conflict	conflict
RECEIVE DIAGNOSTICS	conflict	conflict	conflict	allowed	conflict	conflict
RELEASE(6)/RELEASE(10)	allowed (note)	conflict	conflict	conflict	conflict	conflict
REPORT LUNS	allowed	allowed	allowed	allowed	allowed	allowed
REQUEST SENSE	allowed	allowed	allowed	allowed	allowed	allowed
RESERVE(6)/RESERVE(10)	conflict	conflict	conflict	conflict	conflict	conflict
SEND DIAGNOSTICS	conflict	conflict	conflict	allowed	conflict	conflict
TEST UNIT READY	conflict	conflict	conflict	allowed	conflict	conflict
WRITE BUFFER	conflict	conflict	conflict	allowed	conflict	conflict

Key: LU=Logical Unit, Excl=Exclusive, RO=Registrants Only
Note: The reservation is not released see clause xxx.

Table 2 - PERSISTENT RESERVE OUT service actions that are allowed in the presence of various reservations

PERSISTENT RESERVE OUT service action	Addressed LU is reserved by another initiator (A)	Addressed LU has this type of persistent reservation held by another initiator (B)				
		From any initiator		From Registered Initiator (All RO types)	From Initiator Not Registered	
		Write Excl	Excl Access		Write Excl RO	Excl Access RO
CLEAR	conflict	conflict	conflict	allowed	conflict	conflict
PREEMPT	conflict	conflict	conflict	allowed	conflict	conflict
PREEMPT & ABORT	conflict	conflict	conflict	allowed	conflict	conflict
REGISTER	conflict	allowed	allowed	allowed	allowed	allowed
RELEASE	conflict	conflict	conflict	allowed (note)	conflict	conflict
RESERVE	conflict	conflict	conflict	conflict	conflict	conflict

Key: LU=Logical Unit, Excl=Exclusive, RO=Registrants Only
Note: The reservation is not released see clause 5.3.2.5.1.

The time at which a reservation is established with respect to other tasks being managed by the device server is vendor specific. Successful completion of a reservation command indicates that the new reservation is established. A reservation may apply to some or all tasks queued before the completion of the reservation command. The reservation shall apply to all tasks received by the device server after successful completion of the reservation command. The execution of any reserve/release command or persistent reserve service action shall be performed as a single indivisible event. ~~such that no other command processed by the device server shall be uncertain of the presence of a reservation at the time the device server tests for the presence of a reservation.~~

Multiple reserve/release commands or persistent reserve service actions may be queued at the same time. The order of execution of such commands is defined by the tagged queueing restrictions, if any, but each is executed as a single indivisible ~~complete~~ command without any interleaving of actions that may be required by other reservation commands.

5.3.1 The Reserve/Release management method

The Reserve/Release management method commands, RESERVE(6), RESERVE(10), RELEASE(6), and RELEASE(10) are used among multiple initiators that do not require operations to be protected across initiator failures (and subsequent hard resets). The Reserve/Release reservations management method also allows an application client to provide restricted device access to one additional initiator (a third-party initiator), usually a temporary initiator performing a service for the application client sending the reservation command.

Reservations managed using the Reserve/Release method do not persist across some recovery actions (e.g., hard resets), so most systems require significant reinitialization after a failure that results in a hard reset. Reserve/ Release managed reservations are retained by the device server until released or until reset by mechanisms specified in this standard.

The RESERVE(6) and RESERVE(10) commands allow superseding reservations.

5.3.2 The Persistent Reservations management method

The Persistent Reservations management method is used among multiple initiators that require operations to be protected across initiator failures, which usually involve hard resets. Even though different protocols that transport SCSI handle hard resets differently (e.g., parallel uses a reset signal, fibre channel uses primitive signals) the persistent reservation shall be protected. 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 shall be 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.

Persistent reservations are not reset by the TARGET RESET task management function or other global actions and may, optionally, be preserved across power cycles. Persistent reservations may be used to enforce device sharing among multiple initiators.

The PERSISTENT RESERVE OUT and PERSISTENT RESERVE IN commands provide the basic mechanism for dynamic contention resolution in multiple initiator systems using multiple port targets. Before a persistent reservation may be established an initiator must register with a device server using a reservation key. Reservation keys are necessary to allow:

a) authentication of subsequent PERSISTENT RESERVE OUT commands;
b) identification of other initiators that are registered;
c) identification of the reservation key(s) that have an associated reservation;
d) preemption of a persistent reservation from a failing or uncooperative initiator; and
e) multiple initiators to participate in a reservation.

The reservation key provides a method for the application client to associate a protocol-independent identifier with ~~method to identify~~ an initiator on a specific port of a device server. The reservation key is used in the PERSISTENT RESERVE IN command to identify which initiators are registered, and which initiator, if any, holds the reservation. The reservation key is used in the PERSISTENT RESERVE OUT command; to register an initiator, to verify the initiator issuing the PERSISTENT RESERVATION OUT command is registered, and to specify which initiator's registration or persistent reservation to preempt.

Reservation key values may be used by application clients to identify initiators, using application specific methods that are outside the scope of this standard. This standard provides the ability to register no more than one key per initiator/logical unit pair. Multiple initiators may use the same key for a logical unit. An initiator may establish registrations for multiple logical units in a SCSI device using any combination of unique or duplicate keys. This provides the ability for an application client to preempt multiple initiators with a single PERSISTENT RESERVE OUT command, but it removes the ability for the application client to uniquely identify the initiators using the PERSISTENT RESERVE commands.

5.3.2.1 Preserving persistent reservations

The application client may request the device server to preserve the persistent reservation and registration keys across power cycles by requesting the Activate Persist Through Power Loss (APTPL) capability. The application client may request this as part of registration by setting the APTPL bit to one.

After the application client enables the APTPL capability the device server shall preserve all current and future registrations and persistent reservations associated with the logical unit to which the REGISTER service action was addressed until an application client disables the APTPL capability. The APTPL value from the most recent successfully completed REGISTER service action from any application client shall determine the logical unit's behavior in the event of a power loss. ~~The most recently received valid APTPL value from any application client shall govern the logical unit's behavior in the event of power loss.~~

The device server shall preserve the following information for each registration across any reset, and if the APTPL capability is enabled, across any power off period:

a) Initiator identifier;
b) reservation key; and
c) when supported by the protocol, the initiator port's world wide identification.

The device server shall preserve the following information for the reservation across any reset, and if the APTPL capability is enabled, across any power off period:

a) Initiator identifier;
b) reservation key;
c) scope;
d) type; and
e) when supported by the protocol, the initiator port's world wide identification.

For those protocols for which the initiator port's world wide identification is available to the device server the initiator port's world wide identification shall be used to determine if the initiator identifier has changed. This determination shall be made at any time the target detects that the configuration of the system may have changed. If the initiator identifier changed, the device server shall assign the new initiator identifier to the existing registration and reservation to the initiator port having the same world wide identification.

The capability of preserving persistent reservations and registration keys across power cycles requires the use of a nonvolatile memory within the SCSI device. Any SCSI device that supports the Persist Through Power Loss (APTPL) capability of persistent reservation and has nonvolatile memory that is not ready shall allow ~~accept~~ the following commands into the task set:

a) INQUIRY;
b) LOG SENSE;
c) REPORT LUNS;
d) REQUEST SENSE; and
e) START/STOP UNIT (with start bit = 1 and power condition value of 0).

Any commands, other than those listed above shall return CHECK CONDITION status. The sense key shall be set to NOT READY and the additional sense data shall be set as described in the TEST UNIT READY command (see 7.24).

5.3.2.2 Finding ~~previous~~ persistent reservations and reservation keys

The application client may obtain information about the persistent reservation and the reservation keys that are present within a device server by issuing PERSISTENT RESERVE IN commands with a READ RESERVATION service action or a READ KEYS service action.

5.3.2.2.1 Reporting Reservation keys

An application client may issue a PERSISTENT RESERVE IN command with a service action of READ KEYS to determine if any initiators have registered with a logical unit.

In response to a PERSISTENT RESERVE IN with a READ KEYS service action the device server shall report the following:

a) the current generation value (see clause 7.11.2); and
b) the reservation key for every initiator that is currently registered.

The generation value allows the application client to verify that the configuration of the initiators registered with ~~attached to~~ a logical unit has not been modified. ~~by another application client without the knowledge of the examining application client~~.

~~A reservation key is a value sent to a SCSI device by an initiator executing the PERSISTENT RESERVATION OUT command using a REGISTER service action. The choice of the value for the reservation key for each initiator/ logical unit pair is outside the scope of this standard.~~

Each reservation key may be examined by the application client and correlated with one or more initiators and SCSI ports by mechanisms outside the scope of this standard. Duplicate keys are ~~possible~~ reported if multiple initiators use the same reservation key.

~~An initiator may establish registrations for multiple logical units in a SCSI device using any combination of unique or duplicate keys.~~

5.3.2.2.2 Reporting Persistent reservations

An application client may issue a PERSISTENT RESERVE IN command with a service action of READ RESERVATION to receive the persistent reservation information.

In response to a PERSISTENT RESERVE IN command with a READ RESERVATION service action the device server shall report the following as an uninterrupted series of actions:

a) the current generation value (see clause 7.11.2);
b) the registered reservation key, if any, associated with the initiator that holds the persistent reservation;
c) the scope and type of each persistent reservation, if any; and
d) the scope-specific address, if any.

If an application client ~~using~~ uses ~~unique~~ a different reservation key~~s~~ for each initiator/logical unit pair the application client may use the reservation key to associate the persistent reservation with the initiator that holds the persistent reservation. This association is done using techniques that are outside the scope of this standard.

5.3.2.3 Registering

To establish a persistent reservation the initiator shall first register with a logical unit.

If the initiator has not yet established a reservation key or the reservation key has been removed the registration is accomplished by issuing a PERSISTENT RESERVE OUT command with service action of REGISTER with the following parameters:

a) APTPL bit optionally set to one;
b) reservation key set to zero; and
c) service action reservation key set to a non-zero value.

If the initiator has an established registration it may change its reservation key. This is accomplished by issuing a PERSISTENT RESERVE OUT command with service action of REGISTER with the following parameters:

a) APTPL bit optionally set to one;
b) reservation key set to the value of the previously established reservation key; and
c) service action reservation key set to a non-zero value.

If a PERSISTENT RESERVE OUT with a REGISTER service action 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 INSUFFICIENT

REGISTRATION RESOURCES.

NOTE 1 - It is recommended a target have enough resources to handle a registration from each initiator the target has knowledge of.

In response to a PERSISTENT RESERVE OUT with a REGISTER service action the device server shall perform a registration by doing the following as an uninterrupted series of actions:

a) Process the registration request regardless of any persistent reservations;
b) process the APTPL bit;
c) ignore the contents of the SCOPE and TYPE fields;
d) map the reservation key to the registering initiator using the initiator identification and, if available, the initiator port's world wide identification;
e) register the reservation key without changing any a persistent reservation that may exist; and
f) retain the reservation key and associated information.

After the registration request has been processed the device server shall then allow other PERSISTENT RESERVE OUT commands from the registered initiator to execute.

For each initiator that performs a PERSISTENT RESERVE OUT with a 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 initiator registration is removed (see clause 5.3.2.5).

Any PERSISTENT RESERVE OUT command service action received from an unregistered initiator, other than the REGISTER service action, shall be rejected with a RESERVATION CONFLICT status.

It is not an error for an initiator that is registered to register again with the same reservation key or a new reservation key. A ~~repeated~~ registration shall have no effect on any other registrations (i.e., when more than one initiator is registered with the same reservation key and one of those initiators registers again it has no effect on the other initiator's registrations). A ~~repeated~~ registration, not containing a Service action reservation key of zero, shall have no effect on any reservations (i.e., an initiator may change its reservation key without affecting any previously created reservation).

Multiple initiators may register with the same reservation key. An initiator may use the same reservation key for multiple logical units.

5.3.2.4 Creating a persistent reservation when there is no persistent reservation

An application client creates a persistent reservation by issuing a PERSISTENT RESERVE OUT command with a service action of RESERVE through a registered initiator with the following parameters:

a) Reservation key set to the value of the initiator/logical unit pair's established reservation key; and
b) type and scope set to the reservation being created.

~~Only one persistent reservation at a time per logical unit is allowed. Multiple persistent reservations may be created in SCSI devices that contain multiple elements. However, there shall only be one persistent reservation per element.~~

Only one persistent reservation with a scope of logical unit is allowed at a time per logical unit. Multiple persistent reservations with a scope of element may be created in a logical unit that contains multiple elements. However, there shall only be one persistent reservation per element.

If the target receives a PERSISTENT RESERVE OUT command that attempts to create a persistent reservation

when a persistent reservation already exists for the logical unit from an initiator other than the initiator that created the reservation then the command shall be rejected with a RESERVATION CONFLICT status.

If the initiator that created the persistent reservation attempts to modify the type or scope of an existing reservation then the command shall be rejected with a RESERVATION CONFLICT status.

If the device server receives a PERSISTENT RESERVE OUT command with a service action of RESERVE where the type and scope is the same as the existing type and scope from the initiator that created the persistent reservation it shall not make any change to the existing reservation and shall return a GOOD status.

If the target receives a RESERVE(10) or RESERVE(6) command when a persistent reservation exists for the logical unit then the command shall be rejected with a RESERVATION CONFLICT.

See clause 5.3 for information on when a persistent reservation takes effect.

5.3.2.5 Removing registrations and persistent reservations

A registered initiator using the value of the initiator/logical unit pair's established reservation key may remove a persistent reservation by issuing one of the following commands:

a) a PERSISTENT RESERVE OUT command with a service action of RELEASE from the initiator that performed the reservation (see clause 5.3.2.5.1);
b) a PERSISTENT RESERVE OUT command with a PREEMPT service action specifying the reservation key of the initiator holding the reservation (see clause 5.3.2.5.2.1);
c) a PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action specifying the reservation key of the initiator holding the reservation (see clause 5.3.2.5.2.2); or
d) a PERSISTENT RESERVE OUT command with a service action of CLEAR service action (see clause 5.3.2.5.3).

A registered initiator using the value of the initiator/logical unit pair's established reservation key may remove a registration by issuing one of the following commands:

a) a PERSISTENT RESERVE OUT command with a PREEMPT service action specifying that reservation key (see clause 5.3.2.5.2.1.3);
b) a PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action specifying that reservation key (see clause 5.3.2.5.2.2); or
c) a PERSISTENT RESERVE OUT command with a CLEAR service action (see clause 5.3.2.5.3).
d) a PERSISTENT RESERVE OUT command with a REGISTER service action from the same initiator with the value of the SERVICE ACTION RESERVATION KEY field set to zero.

When a reservation key has been removed, no information shall be reported for that unregistered initiator in subsequent READ KEYS service action(s). Any persistent reservation associated with that unregistered initiator shall be released. If that released persistent reservation was of the type write exclusive-registrants only or exclusive access-registrants only the device server shall establish a unit attention condition for all the other registered initiators. The sense key shall be set to UNIT ATTENTION and the additional sense data shall be set to RESERVATIONS RELEASED.

A persistent reservation may also be released by a power off if the APTPL capability is not enabled. When the most recent APTPL value received by the device server is zero (see 7.12.2), a power off/on cycle:

a) performs a hard reset;
b) clears all persistent reservations; and
c) removes all registered reservation keys (see 5.3.2.3).

5.3.2.5.1 Releasing a persistent reservation

Only the initiator that creates the persistent reservation is allowed to release that persistent reservation regardless of the type of persistent reservation.

An application client releases a persistent reservation it holds by issuing a PERSISTENT RESERVE OUT command with a service action of RELEASE through the registered initiator that holds the persistent reservation with the following parameters:

a) Reservation key set to the value of the initiator/logical unit pair's established reservation key; and
b) type and scope set to match the persistent reservation being released.

In response to a persistent reservation release request from an initiator that created the persistent reservation the device server shall perform a release by doing the following as an uninterrupted series of actions:

a) Removing the persistent reservation;
b) not removing any registration key(s);
c) If the released persistent reservation is of the type write exclusive-registrants only or exclusive access-registrants only the device server shall establish a unit attention condition for all the other registered initiators. The sense key shall be set to UNIT ATTENTION and the additional sense data shall be set to RESERVATION RELEASED; and

Editors Note 1 - GOP: RESERVATION RELEASED is a new ASCQ. This falls under the other note about when reservation go away. (2Ah 04h)

d) If the persistent reservation is of any other type the device server shall not establish an unit attention condition.

The device server shall return a CHECK CONDITION status for a PERSISTENT RESERVE OUT command that specifies the release of a persistent reservation held by the requesting initiator if the scope and type does not match the scope and type of the established persistent reservation. The sense key shall be set to ILLEGAL REQUEST and additional sense data shall be set to INVALID RELEASE OF PERSISTENT RESERVATION.

Editors Note 2 - GOP: INVALID RELEASE OF ACTIVE PERSISTENT RESERVATION needs to be changed to INVALID RELEASE OF PERSISTENT RESERVATION

In response to a persistent reservation release request from a registered initiator using the value of the initiator/ logical unit pair's established reservation key that does not hold the persistent reservation or if there is no persistent reservation the device server shall do the following:

a) Not remove the persistent reservation (if any);
b) not remove or change any registration key(s); and
c) return a status of GOOD.

5.3.2.5.2 Preempting an existing persistent reservation

5.3.2.5.2.1 PREEMPT service action

A PERSISTENT RESERVE OUT command with a PREEMPT service action is used to preempt a persistent reservation and/or registration. The determination of whether the preempt relates to a persistent reservation or a

registration is made by the device server by examining the service action reservation key of the PREEMPT service action. If the service action reservation key is associated with the reservation being preempted then the reservation is preempted and any matching registration(s) removed (see clause 5.3.2.5.2.1.2). If the service action reservation key is not associated with the reservation being preempted then any matching registration(s) are removed (see clause 5.3.2.5.2.1.3).

See figure 1 for a description of how a device server should interpret a PREEMPT service action to determine the actions it should take (e.g., preempt persistent reservation, preempt registration or preempt both registration and persistent reservation).

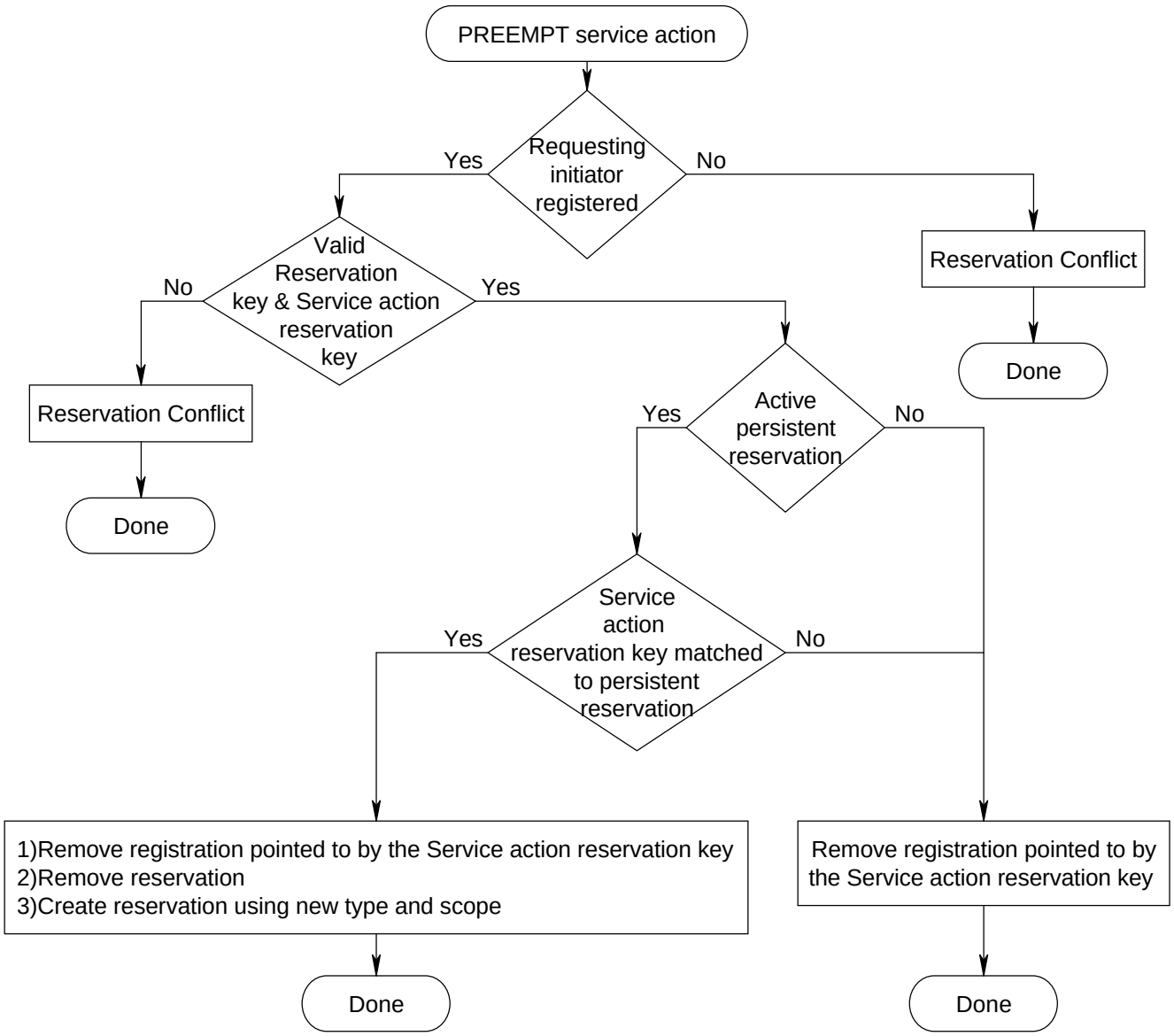

Figure 1 - Device server interpretation of PREEMPT service action

5.3.2.5.2.1.1 Failed persistent reservation preempt

If the preempting initiator's PREEMPT service action or PREEMPT AND ABORT service action fails (e.g., queue full, busy, time-out due to queueing restrictions or time-out due to queue blocked due to failed initiator) it may issue a Logical Unit Reset task management function to the failing logical unit to remove blocking tasks and then reissue the preempting service action.

5.3.2.5.2.1.2 Preempting reservations

Any registered initiator may preempt any persistent reservation with another persistent reservation by issuing a PERSISTENT RESERVE OUT command with a PREEMPT service action through a registered initiator with the following parameters:

a) Reservation key set to the value of the initiator/logical unit pair's established reservation key;
b) service action reservation key set to match the reservation key of the persistent reservation being preempted; and
c) type and scope set to define a new persistent reservation. The scope and type of the persistent reservation created by preempting initiator may be different than the persistent reservation being preempted.

If the Service action reservation key is associated with a reservation the device server shall perform a preempt by doing the following as an uninterrupted series of actions:

a) remove the persistent reservation for the initiator identified by the Service action reservation key specified in the PERSISTENT RESERVE OUT parameter list;
b) remove the registration for the initiator or initiators identified by the Service action reservation key specified in the PERSISTENT RESERVE OUT parameter list (see 5.3.2.3);
c) establish a persistent reservation for the preempting initiator;
d) process tasks as defined in clause 5.3; and
e) establish a unit attention condition for any initiator that lost its reservation and/or registration. The sense key shall be set to UNIT ATTENTION and the additional sense data shall be set to REGISTRATIONS PREEMPTED.

Tasks received after successful completion of the PERSISTENT RESERVE OUT command with the PREEMPT service action are subject to the persistent reservation restrictions established by the preempting initiator. Each task in the dormant, blocked, or enable state at the time the PERSISTENT RESERVE OUT command with the PREEMPT service action is received shall be subject in a vendor specific manner to either the restrictions established by the persistent reservation being preempted or to the restrictions established by the preempting initiator. Completion status shall be returned for each task.

A PERSISTENT RESERVE OUT specifying a PREEMPT service action with the SERVICE ACTION RESERVATION KEY with a value equal to the reservation key is not an error. In that case the device server shall establish the new reservation.

5.3.2.5.2.1.3 Preempting registrations

An application client may clear registrations without affecting a persistent reservation by issuing a PERSISTENT RESERVE OUT command with a PREEMPT service action through a registered initiator with the following parameters:

a) Reservation key set to the value of the initiator/logical unit pair's established reservation key; and
b) service action reservation key set to match the reservation key of the registration being cleared.

If the Service action reservation key is not associated with a reservation the device server shall perform a preempt

by doing the following in an uninterrupted series of actions:

a) remove the registration for the initiator or initiators identified by the Service action reservation key specified in the PERSISTENT RESERVE OUT parameter list;
b) ignore the contents of the SCOPE and TYPE fields;
c) process tasks as defined in clause 5.3; and
d) establish a unit attention condition for any initiator that lost its registration. The sense key shall be set to UNIT ATTENTION and the additional sense data shall be set to REGISTRATIONS PREEMPTED.

Editors Note 3 - GOP: REGISTRATIONS PREEMPTED is a new ASC/ASCQ. (2Ah 05h)

If a PERSISTENT RESERVE OUT specifying a PREEMPT service action sets the Service action reservation key to a value that does not match any registered reservation key the device server shall return a RESERVATION CONFLICT status.

5.3.2.5.2.2 PREEMPT AND ABORT service action

The initiator's request for and the device server's responses to a PERSISTENT RESERVE OUT command PREEMPT AND ABORT service action are identical to the PREEMPT service action (clause 5.3.2.5.2.1) except for the following additions. If no reservation conflict occurred, the device server shall do the following as part of the uninterrupted series of actions.

a) Every task from all preempted initiators shall be terminated as if an ABORT TASK SET task management function had been performed by all of the preempted initiators. After GOOD status has been returned for the PERSISTENT RESERVE OUT command, new tasks are subject to the persistent reservation restrictions established by the preempting initiator.
b) The device server shall clear any ACA or CA condition associated with an initiator being preempted and shall clear any tasks with an ACA attribute from that initiator. If TST=000b (see 8.3.4), then ACA or CA conditions for initiators, other than the initiator being preempted, shall prevent the execution of the PERSISTENT RESERVE OUT command which shall end with a status of ACA ACTIVE if NACA=1 (see SAM) or BUSY if NACA=0. If TST=001b, then ACA or CA conditions for initiators other than the initiator being preempted shall not prevent the execution of the PERSISTENT RESERVE OUT task.
c) For SCSI devices that implement the PREVENT ALLOW MEDIUM REMOVAL command, the device server shall perform an action equivalent to the execution of a PREVENT ALLOW MEDIUM REMOVAL command with the PREVENT field equal to zero for the initiator or initiators being preempted (see clause 7.13).

Any asynchronous event reporting operations in progress are not affected by the PREEMPT AND ABORT service action.

5.3.2.5.3 Clearing a persistent reservation

Any application client may clear a persistent reservation and all registrations from a device server for a specific logical unit by issuing a PERSISTENT RESERVE OUT command with a service action of CLEAR service action through a registered initiator with the following parameter:

a) Reservation key set to the value of the initiator/logical unit pair's established reservation key.

In response to this request the device server shall perform a clear by doing the following as part of an uninterrupted series of actions:

a) Remove any persistent reservation associated with the logical unit;

b) remove all registration key(s) (see 5.3.2.3);
c) ignore the contents of the SCOPE and TYPE fields;
d) continue normal execution of any tasks from any initiator that have been accepted by the device server as nonconflicting; and
e) establish a unit attention condition for all registered initiators, if any, for the cleared logical unit. The sense key shall be set to UNIT ATTENTION and the additional sense data shall be set to RESERVATION PREEMPTED.

Editors Note 4 - GOP: RESERVATION PREEMPTED is a new ASCQ of 2Ah 03h.

Application clients should not use the CLEAR service action except during recoveries that are associated with initiator or system reconfiguration, since the effect of the CLEAR service action is to remove the persistent reservation management conventions that protect data integrity.

7.11 PERSISTENT RESERVE IN command

The PERSISTENT RESERVE IN command (see table 3) is used to obtain information about persistent reservations and reservation keys that are present within a device server. This command is used in conjunction with the PERSISTENT RESERVE OUT command (see 7.12).

Table 3 - PERSISTENT RESERVE IN command

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (5Eh)							
1	Reserved			Service action				
2	RESERVED							
3	RESERVED							
4	RESERVED							
5	RESERVED							
6	RESERVED							
7	(MSB)			Allocation length				
8								(LSB)
9	CONTROL							

The actual length of the PERSISTENT RESERVE IN parameter data is available in the parameter data. The ALLOCATION LENGTH field in the CDB indicates how much space has been reserved for the returned parameter list. If the length is not sufficient to contain the entire parameter list, the first portion of the list shall be returned. This shall not be considered an error. If the remainder of the list is required, the application client should send a new PERSISTENT RESERVE IN command with a ALLOCATION LENGTH field large enough to contain the entire list.

7.11.1 PERSISTENT RESERVE IN Service Actions

The SERVICE ACTION codes for the PERSISTENT RESERVE IN command are defined in table 4.

Table 4 - PERSISTENT RESERVE IN SERVICE ACTION codes

Code	Name	Description
00h	READ KEYS	Reads all registered Reservation Keys
01h	READ RESERVATION	Reads the current persistent reservation
02h - 1Fh	Reserved	Reserved

7.11.1.1 Read Keys

The READ KEYS service action requests that the device server return a parameter list containing a header and a list of each currently registered initiator's reservation key. If multiple initiators have registered with the same key, then that key value shall be listed multiple times, once for each such registration.

For more information on read keys see clause 5.3.2.2.1.

7.11.1.2 Read Reservation

The Read Reservation service action requests that the device server return a parameter list containing a header and the persistent reservation, if any, that is present in the device server.

For more information on read reservation see clause 5.3.2.2.2.

7.11.2 PERSISTENT RESERVE IN parameter data for Read Keys

The format for the parameter data provided in response to a PERSISTENT RESERVE IN command with the Read Keys service action is shown in table 5.

Table 5 - PERSISTENT RESERVE IN parameter data for Read Keys

Bit Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				Generation				(LSB)
4	(MSB)							
7				ADDITIONAL LENGTH (n-7)				(LSB)
				Reservation key list				
8	(MSB)							
15				RESERVATION KEY (first)				(LSB)
				.				
				.				
				.				
n-7	(MSB)							
n				RESERVATION KEY (last)				(LSB)

The GENERATION field contains a 32-bit counter that the device server shall increment every time a PERSISTENT RESERVE OUT command requests a REGISTER, a CLEAR, a PREEMPT, or a PREEMPT AND ABORT service action. The counter shall not be incremented by a PERSISTENT RESERVE IN command, by a PERSISTENT RESERVE OUT command that performs a RESERVE or RELEASE service action, or by a PERSISTENT RESERVE OUT command that is not performed due to an error or reservation conflict. Regardless of the APTPL value the generation value shall be set to 0 as part of the power on reset process.

The ADDITIONAL LENGTH field contains a count of the number of bytes in the Reservation key list. If the allocation length specified by the PERSISTENT RESERVE IN command is not sufficient to contain the entire parameter list, then only the bytes from 0 to the maximum allowed allocation length shall be sent to the application client. The incremental remaining bytes shall be truncated, although the ADDITIONAL LENGTH field shall still contain the actual number of bytes in the reservation key list without consideration of any truncation resulting from an insufficient allocation length. This shall not be considered an error.

The RESERVATION KEY list contains the 8-byte reservation keys for all initiators that have registered through all ports with the device server.

7.11.3 PERSISTENT RESERVE IN parameter data for Read Reservation

The format for the parameter data provided in response to a PERSISTENT RESERVE IN command with the Read Reservation service action is shown in table 6.

Table 6 - PERSISTENT RESERVE IN parameter data for Read Reservations

Bit Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				Generation				(LSB)
4	(MSB)							
7				ADDITIONAL LENGTH (n-7)				(LSB)
8	(MSB)							
n				RESERVATION DESCRIPTORS				(LSB)

The GENERATION field shall be as defined for the PERSISTENT RESERVE IN Read Keys parameter data (see 7.11.2).

The ADDITIONAL LENGTH field contains a count of the number of bytes to follow in the RESERVATION DESCRIPTOR(s). If the allocation length specified by the PERSISTENT RESERVE IN command is not sufficient to contain the entire parameter list, then only the bytes from 0 to the maximum allowed allocation length shall be sent to the application client. The remaining bytes shall be truncated, although the ADDITIONAL LENGTH field shall still contain the actual number of bytes of the RESERVATION DESCRIPTOR(s) and shall not be affected by the truncation. This shall not be considered an error.

The format of the RESERVATION DESCRIPTORS is defined in table 7. There shall be a RESERVATION DESCRIPTOR for the persistent reservation, if any, present in the logical unit and a RESERVATION DESCRIPTOR for each element, if any, having a persistent reservation.

Table 7 - PERSISTENT RESERVE IN Reservation Descriptor

Bit Byte	7	6	5	4	3	2	1	0
0	(MSB)							
7				Reservation key				(LSB)
8	(MSB)							
11				Scope-specific address				(LSB)
12	RESERVED							
13	Scope				Type			
14	RESERVED							
15	RESERVED							

If a persistent reservation is present in the logical unit that does not contain elements, there shall be a single RESERVATION DESCRIPTOR in the list of parameter data returned by the device server in response to the PERSISTENT RESERVE IN command with a READ RESERVATION service action. The RESERVATION DESCRIPTOR for each reservation shall contain the RESERVATION KEY under which the persistent reservation is held. The type and scope of each persistent reservation as present in the PERSISTENT RESERVE OUT command that created the persistent reservation shall be returned (see 7.11.3.1 and 7.11.3.2).

If a persistent reservation is present in the logical unit that does contain elements, there shall be a RESERVATION DESCRIPTOR in the list of parameter data returned by the device server in response to the PERSISTENT RESERVE IN command with a READ RESERVATION service action for the LU persistent reservation that is held, if any, and each element persistent reservation that may be held. The RESERVATION DESCRIPTOR shall contain the RESERVATION KEY under which the persistent reservation is held. The type and scope of the persistent reservation as present in the PERSISTENT RESERVE OUT command that created the persistent reservation shall be returned (see 7.11.3.1 and 7.11.3.2).

If the scope is an element reservation, the SCOPE-SPECIFIC ADDRESS field shall contain the element address, zero filled in the most significant bytes to fit the field. If the scope is a logical unit reservation the SCOPE-SPECIFIC ADDRESS shall be set to zero.

7.11.3.1 Persistent Reservations Scope

The value in the SCOPE field shall indicate whether a persistent reservation applies to an entire logical unit or to an element. The values in the SCOPE field are defined in table 8.

Table 8 - Persistent Reservation SCOPE Codes

Code	Name	Description
0h	LU	Persistent reservation applies to the full logical unit
1h		Obsolete
2h	Element	Persistent reservation applies to the specified element
3h - Fh	Reserved	Reserved

7.11.3.1.1 LU Scope

A SCOPE field value of LU shall indicate that the persistent reservation applies to the entire logical unit. The LU scope shall be implemented by all device servers that implement PERSISTENT RESERVE OUT.

7.11.3.1.2 Element Scope

A SCOPE field value of element shall indicate that the persistent reservation applies to the element of the logical unit defined by the SCOPE-SPECIFIC ADDRESS field in the PERSISTENT RESERVE OUT parameter list. An element is defined by the SCSI-3 Medium Changer Commands (SMC) standard. The Element scope is optional for all device servers that implement PERSISTENT RESERVE OUT.

7.11.3.2 Persistent Reservations Type

The value in the TYPE field shall specify the characteristics of the persistent reservation being established for all data blocks within the element or within the logical unit. Table 9 defines the characteristics of the different type values. For each persistent reservation type, table 9 lists the code value and describes the required device server support. In table 9, the description of required device server support is divided into two paragraphs. The first paragraph defines the required handling for read operations. The second paragraph defines the required handling

for write operations.

Table 9 - Persistent Reservation Type Codes

Code	Name	Description
0h		Obsolete
1h	Write Exclusive	**Reads Shared**: Any application client on any initiator may execute tasks that perform transfers from the storage medium or cache of the logical unit to the initiator. **Writes Exclusive**: Any task from any initiator other than the initiator holding the persistent reservation that requests a transfer from the initiator to the storage medium or cache of the logical unit shall result in a reservation conflict.
2h		Obsolete
3h	Exclusive Access	**Reads Exclusive**: Any task from any initiator other than the initiator holding the persistent reservation that requests a transfer from the storage medium or cache of the logical unit to the initiator shall result in a reservation conflict. **Writes Exclusive**: Any task from any initiator other than the initiator holding the persistent reservation that requests a transfer from the initiator to the storage medium or cache of the logical unit shall result in a reservation conflict.
4h		Obsolete
5h	Write Exclusive-registrants Only	**Reads Shared**: Any application client on any initiator may execute tasks that request transfers from the storage medium or cache of the logical unit to the initiator. **Writes Exclusive**: A task that requests a transfer to the storage medium or cache of the logical unit from an initiator that ~~has not previously requested a REGISTER service action with~~ is not currently registered with the device server shall result in a reservation conflict.
6h	Exclusive Access-Registrants Only	**Reads Exclusive**: A task that requests a transfer from the storage medium or cache of the logical unit to an initiator that has not previously requested a REGISTER service action with the device server shall result in a reservation conflict. **Writes Exclusive**: A task that requests a transfer to the storage medium or cache of the logical unit from an initiator that ~~has not previously requested a REGISTER service action with~~ is not currently registered with the device server shall result in a reservation conflict
7h - Fh	Reserved	

7.12 PERSISTENT RESERVE OUT command

The PERSISTENT RESERVE OUT command (see table 10) is used to request service actions that reserve a logical unit or element for the exclusive or shared use of a particular initiator. The command uses other service actions to manage and remove such reservations. The command shall be used in conjunction with the PERSISTENT RESERVE IN command and shall not be used with the RESERVE and RELEASE commands.

Initiators requesting PERSISTENT RESERVE OUT service actions are identified by a reservation key provided by the application client. An application client may use the PERSISTENT RESERVE IN command to identify which initiators is holding a persistent reservation and use the PERSISTENT RESERVE OUT command to preempt that reservation if required.

Table 10 - PERSISTENT RESERVE OUT command

Bit Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (5Fh)							
1	Reserved				Service action			
2	Scope				Type			
3	RESERVED							
4	RESERVED							
5	RESERVED							
6	RESERVED							
7	(MSB)			PARAMETER LIST LENGTH (18h)				
8								(LSB)
9	CONTROL							

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 INSUFFICIENT REGISTRATION RESOURCES.

Editors Note 5 - GOP: Change the INSUFFICIENT RESERVATION RESOURCES ASCQ to INSUFFICIENT REGISTRATION RESOURCES. (55h 02h)

The PERSISTENT RESERVE OUT command contains fields that specify a persistent reservation service action, the intended scope of the persistent reservation, and the restrictions caused by the persistent reservation. The TYPE and SCOPE fields are defined in 7.11.3.1 and 7.11.3.2. If a SCOPE field specifies a scope that is not implemented, the device server shall return a CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and additional sense data shall be set to INVALID FIELD IN CDB.

Fields contained in the PERSISTENT RESERVE OUT parameter list specify the information required to perform a particular persistent reservation service action.

The parameter list shall be 24 bytes in length and the Parameter list length field shall contain 24 (18h). If the

Parameter list length is not 24, 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 PARAMETER LIST LENGTH ERROR.

7.12.1 PERSISTENT RESERVE OUT Service Actions

When processing the PERSISTENT RESERVE OUT service actions, the device server shall process the generation value as specified in 7.11.2.

The PERSISTENT RESERVE OUT command service actions are defined in table 11.

Table 11 - PERSISTENT RESERVE OUT service action codes

Code	Name	Description
00h	REGISTER	Registers a reservation key with the device server (for more information on registering see clause 5.3.2.3).
01h	RESERVE	Creates a persistent reservation having a specified scope and type. The scope and type of a persistent reservation are defined in 7.11.3.1 and 7.11.3.2 (for more information on reserving see clause 5.3.2.4).
02h	RELEASE	Releases the selected reservation for the requesting initiator (for more information on releasing see clause 5.3.2.5.1).
03h	CLEAR	Clears all reservation keys and all persistent reservations (for more information on clearing see clause 5.3.2.5.3).
04h	PREEMPT	Preempts persistent reservations from another initiator (for more information on preempting see clause 5.3.2.5.2.1).
05h	PREEMPT & ABORT	Preempts persistent reservations from another initiator and aborts the task set for the preempted initiator (for more information on preempting and aborting see clause 5.3.2.5.2.1 and clause 5.3.2.5.2.2).
06h - 1Fh	Reserved	

The parameter list values for each service action are specified in clause 7.12.2

7.12.2 PERSISTENT RESERVE OUT parameter list

The parameter list required to perform the PERSISTENT RESERVE OUT command is defined in table 12. All fields shall be sent on all PERSISTENT RESERVE OUT commands, even if a particular field is not required for the

specified SERVICE ACTION and scope values.

Table 12 - PERSISTENT RESERVE OUT parameter list

Bit Byte	7	6	5	4	3	2	1	0
0	(MSB)							
7				Reservation key				(LSB)
8	(MSB)							
15				Service action Reservation key				(LSB)
16	(MSB)							
19				Scope-specific address				(LSB)
20				RESERVED				APTPL
21				RESERVED				
21				RESERVED				
21				RESERVED				

The RESERVATION KEY field contains an 8-byte value provided by the application client to the device server to identify the initiator that is the source of the PERSISTENT RESERVE OUT command. The device server shall verify that the RESERVATION KEY field in a PERSISTENT RESERVE OUT command matches the registered reservation key for the initiator from which the task was received except for the Register service action for an unregistered initiator which shall have a reservation key value of zero. If a PERSISTENT RESERVE OUT command specifies a RESERVATION KEY field other than the reservation key registered for the initiator, the device server shall return a RESERVATION CONFLICT status. The reservation key of the initiator shall be verified to be correct regardless of the service action and scope values.

The SERVICE ACTION RESERVATION KEY field contains information needed for the following service actions; the Register, Preempt, and Preempt and Abort service actions. For the Register service action, the SERVICE ACTION RESERVATION KEY field contains the new reservation key to be registered. For the Preempt and Preempt and Abort service actions, the SERVICE ACTION RESERVATION KEY field contains the reservation key of the persistent reservations that are being preempted. The service action reservation key is ignored for all other service actions.

If the scope is an Element reservation, the scope-specific address field shall contain the Element address, zero filled in the most significant bytes to fit the field. If the service action is Register or Clear or if the scope is a logical unit reservation, the SCOPE-SPECIFIC ADDRESS field shall be set to zero.

The Activate Persist Through Power Loss (APTPL) bit shall be valid only for the Register service action. In all other cases, the APTPL bit shall be ignored. Support for an APTPL bit equal to one is optional. If a device server that does not support the APTPL bit value of one receives that value in a Register service action, the device server shall return a CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and additional sense data shall be set to INVALID FIELD IN PARAMETER LIST.

If the last valid APTPL bit value received by the device server is zero, the loss of power in the target shall release the persistent reservation for all logical units and remove all reservation keys (see 5.3.2.3). If the last valid APTPL bit value received by the device server is one, the logical unit shall retain any persistent reservation that may be present and the reservation keys for all registered initiators even if power is lost and later returned.

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.

Table 13 - PERSISTENT RESERVE OUT service actions and valid parameters

Service Action	Allowed Scope	Parameters		
		Type	Service Action Reservation key	Element or Element Parameters
Register	ignored	ignored	valid	ignored
Reserve Reserve	LU Element	valid valid	ignored ignored	ignored Element valid
Release Release	LU Element	valid valid	ignored ignored	ignored Element valid
Clear	ignored	ignored	ignored	ignored
Preempt Preempt	LU Element	valid valid	valid valid	ignored Element valid
Preempt & abort Preempt & abort	LU Element	valid valid	valid valid	ignored Element valid

7.13 PREVENT ALLOW MEDIUM REMOVAL command

In the description of the PREVENT ALLOW MEDIUM REMOVAL command, section 7.13 of SPC2r05.pdf, pdf page 83, there shall be a new paragraph added after the list. The wording would then become:

The prevention of medium removal shall begin when any application client issues a PREVENT ALLOW MEDIUM REMOVAL command with a Prevent bit of one (medium removal prevented). The prevention of medium removal for the logical unit shall terminate:

a) after all initiators with application clients that previously prevented medium removal issue PREVENT ALLOW MEDIUM REMOVAL commands with a Prevent bit of zero, and the device server has successfully performed a synchronize cache operation; or
b) upon a hard reset condition.

For an initiator that has executed a PERSISTENT RESERVE OUT command with a service action of RESERVE or REGISTER, the Prevent field shall be set to zero as part of the uninterrupted sequence of events performed by a PERSISTENT RESERVE OUT command with a service action of PREEMPT AND ABORT using that initiator's registration value in the Service Action Reservation Key field. This allows an initiator to override the prevention of medium removal function for an initiator that is no longer operating correctly.

Annex A

(informative)

Commands allowed in the presence of various reservations

A.1 SBC commands

This clause should be placed into the model clause of the next version of the SBC standard when, and if, a new version of that standard is published. It should replace all the individual command descriptions of how reservations work.

Reservation restrictions are placed on commands as a result of access qualifiers associated with the type of reservation. The details of which commands are allowed under what types of reservations are described in table A.1. For the reservation restrictions placed on commands for the Reserve/Release management method see table A.1 column A. For the reservation restrictions placed on commands for the Persistent Reservations management method, see table A.1 columns under B.

In table A.1, table A.2, and table A.3 the following key words are used:

allowed: Commands issued by initiators not holding the reservation or by initiators not registered when a registrants only persistent reservation is present should complete normally.

conflict: Commands issued by initiators not holding the reservation or by initiators not registered when a registrants only persistent reservation is present shall not be performed and the device server shall terminate the command with a RESERVATION CONFLICT status.

Commands from initiators holding a reservation should complete normally. The behavior of commands from registered initiators when a registrants only persistent reservation is present is specified in table A.1, table A.2, and table A.3.

A command that does not explicitly write the medium shall be checked for reservation conflicts before the command enters the current task state for the first time. Once the command has entered the current task state, it shall not be terminated with a RESERVATION CONFLICT due to a subsequent reservation.

A command that explicitly writes the medium shall be checked for reservation conflicts before the device server modifies the medium or cache as a result of the command. Once the command has modified the medium, it shall not be terminated with a RESERVATION CONFLICT due to a subsequent reservation.

For each command, this standard or a related command standard defines the conditions that result in RESERVATION CONFLICT. Depending on the particular command standard the conditions are defined in that standard's device model clause or in the clauses that define the specific commands. Annex A contains the RESERVATION CONFLICT information for some of the command sets.

Table A.1 - SBC direct access commands that are allowed in the presence of various reservations

Command	Addressed LU is reserved by another initiator (A)	Addressed LU has this type of persistent reservation held by another initiator (B)				
		From any initiator		From Registered Initiator (All RO types)	From Initiator Not Registered	
		Write Excl	Excl Access		Write Excl RO	Excl Access RO
FORMAT UNIT	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
LOCK/UNLOCK CACHE	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
PRE-FETCH	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
READ(6)/READ(10)/READ(12)	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
READ CAPACITY	Allowed	Allowed	Allowed	Allowed	Allowed	Allowed
READ DEFECT DATA(10)/READ DEFECT DATA(12)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
READ LONG	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
REASSIGN BLOCKS	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
REBUILD	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
REGENERATE	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
SEEK(10)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
SET LIMITS(10)/SET LIMITS(12)	Allowed	Allowed	Allowed	Allowed	Allowed	Allowed
START/STOP UNIT START=1 and POWER CONDITION=0	Allowed	Allowed	Allowed	Allowed	Allowed	Allowed
START/STOP UNIT START = 0 or POWER CONDITION<>0	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
SYNCHRONIZE CACHE	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
VERIFY	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
WRITE(6)/WRITE(10)/WRITE(12)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE AND VERIFY	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE LONG	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE SAME	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
XDREAD	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
XDWRITE	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
XDWRITE EXTENDED	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
XPWRITE	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict

Key: LU=Logical Unit, Excl=Exclusive, RO= Registrants Only

Table A.2 - SBC optical memory commands that are allowed in the presence of various reservations

Command	Addressed LU is reserved by another initiator (A)	Addressed LU has this type of persistent reservation held by another initiator (B)				
		From any initiator		From Registered Initiator (All RO types)	From Initiator Not Registered	
		Write Excl	Excl Access		Write Excl RO	Excl Access RO
ERASE(10)/ERASE(12)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
FORMAT UNIT	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
LOCK/UNLOCK CACHE	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
MEDIUM SCAN	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
PRE-FETCH	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
READ(6)/READ(10)/READ(12)	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
READ CAPACITY	Allowed	Allowed	Allowed	Allowed	Allowed	Allowed
READ DEFECT DATA(10)/READ DEFECT DATA(12)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
READ GENERATION	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
READ LONG	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
READ UPDATED BLOCK	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
REASSIGN BLOCKS	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
SEEK(10)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
SET LIMITS(10)/SET LIMITS(12)	Allowed	Allowed	Allowed	Allowed	Allowed	Allowed
START/STOP UNIT START=1 and POWER CONDITION=0	Allowed	Allowed	Allowed	Allowed	Allowed	Allowed
START/STOP UNIT START = 0 or POWER CONDITION<>0	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
SYNCHRONIZE CACHE	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
UPDATE BLOCK	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
VERIFY(10)/VERIFY(12)	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
WRITE(6)/WRITE(10)/WRITE(12)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE AND VERIFY(10)/WRITE AND VERIFY(12)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE LONG	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict

Key: LU=Logical Unit, Excl=Exclusive, RO= Registrants Only

Table A.3 - SBC write-once commands that are allowed in the presence of various reservations

Command	Addressed LU is reserved by another initiator (A)	Addressed LU has this type of persistent reservation held by another initiator (B)				
		From any initiator		From Registered Initiator (All RO types)	From Initiator Not Registered	
		Write Excl	Excl Access		Write Excl RO	Excl Access RO
LOCK/UNLOCK CACHE	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
MEDIUM SCAN	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
PRE-FETCH	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
READ(6)/READ(10)/READ(12)	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
READ CAPACITY	Allowed	Allowed	Allowed	Allowed	Allowed	Allowed
READ LONG	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
REASSIGN BLOCKS	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
SEEK(10)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
SET LIMITS(10)/SET LIMITS(12)	Allowed	Allowed	Allowed	Allowed	Allowed	Allowed
START/STOP UNIT START=1 and POWER CONDITION=0	Allowed	Allowed	Allowed	Allowed	Allowed	Allowed
START/STOP UNIT START = 0 or POWER CONDITION<>0	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
SYNCHRONIZE CACHE	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
VERIFY(10)/VERIFY(12)	Conflict	Allowed	Conflict	Allowed	Allowed	Conflict
WRITE(6)/WRITE(10)/WRITE(12)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE AND VERIFY(10)/WRITE AND VERIFY(12)	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE LONG	Conflict	Conflict	Conflict	Allowed	Conflict	Conflict
Key: LU=Logical Unit, Excl=Exclusive, RO= Registrants Only						

A.2 SMC commands

This clause should be placed into the model clause of the next version of the SMC standard when, and if, a new version of that standard is published. It should replace all the individual command descriptions of how reservations work.

Reservation restrictions are placed on commands as a result of access qualifiers associated with the type of reservation. The details of which commands are allowed under what types of reservations are described in table A.4. For the reservation restrictions placed on commands for the Reserve/Release management method see table A.4 column A. For the reservation restrictions placed on commands for the Persistent Reservations management method, see table A.4 columns under B.

IIn table A.4 the following key words are used:

allowed: Commands issued by initiators not holding the reservation or by initiators not registered when a registrants only persistent reservation is present should complete normally.

conflict: Commands issued by initiators not holding the reservation or by initiators not registered when a registrants only persistent reservation is present shall not be performed and the device server shall terminate the command with a RESERVATION CONFLICT status.

Commands from initiators holding a reservation should complete normally. The behavior of commands from registered initiators when a registrants only persistent reservation is present is specified in table A.4.

A command that does not explicitly write the medium shall be checked for reservation conflicts before the command enters the current task state for the first time. Once the command has entered the current task state, it shall not be terminated with a RESERVATION CONFLICT due to a subsequent reservation.

A command that explicitly writes the medium shall be checked for reservation conflicts before the device server modifies the medium or cache as a result of the command. Once the command has modified the medium, it shall not be terminated with a RESERVATION CONFLICT due to a subsequent reservation.

For each command, this standard or a related command standard defines the conditions that result in RESERVATION CONFLICT. Depending on the particular command standard the conditions are defined in that standard’s device model clause or in the clauses that define the specific commands. Annex A contains the RESERVATION CONFLICT information for some of the command sets.

Table A.4 - SMC commands that are allowed in the presence of various reservations

Command	Addressed LU is reserved by another initiator (A)	Addressed LU has this type of persistent reservation held by another initiator (B)				
		From any initiator		From Registered Initiator (All RO types)	From Initiator Not Registered	
		Write Excl	Excl Access		Write Excl RO	Excl Access RO
EXCHANGE MEDIUM	conflict	conflict	conflict	allowed	conflict	conflict
INITIALIZE ELEMENT	conflict	conflict	conflict	allowed	conflict	conflict
MOVE MEDIUM	conflict	conflict	conflict	allowed	conflict	conflict
MOVE MEDIUM ATTACHED	conflict	conflict	conflict	allowed	conflict	conflict
POSITION TO ELEMENT	conflict	conflict	conflict	allowed	conflict	conflict
READ ELEMENT STATUS CURDATA=0	conflict	conflict	conflict	allowed	conflict	conflict
READ ELEMENT STATUS CURDATA=1	allowed	allowed	allowed	allowed	allowed	allowed
READ ELEMENT STATUS ATTACHED CURDATA=0	conflict	conflict	conflict	allowed	conflict	conflict
READ ELEMENT STATUS ATTACHED CURDATA=1	allowed	allowed	allowed	allowed	allowed	allowed
RELEASE ELEMENT(6)	allowed	conflict	conflict	conflict	conflict	conflict
READ RELEASE ELEMENT(10)	allowed	conflict	conflict	conflict	conflict	conflict
REQUEST VOLUME ELEMENT ADDR	conflict	conflict	conflict	allowed	conflict	conflict
RESERVE ELEMENT (6)	conflict	conflict	conflict	conflict	conflict	conflict
RESERVE ELEMENT(10)	conflict	conflict	conflict	conflict	conflict	conflict
SEND VOLUME TAG	conflict	conflict	conflict	allowed	conflict	conflict

Key: LU=Logical Unit, Excl=Exclusive, RO= Registrants Only