03-232r1 Modified Reservation Handling proposal

Date: November 4, 2003

To: T10 Committee (SCSI)

From: Roger Cummings (VERITAS)

Subject: T10/03-232r1 SPC-3 Proposal for Modified Reservation Handling

Revision History

03-232r0 (July 2, 2003) First Revision
03-232r1 (November 4, 2003) Added new rows and footnotes to Table 31 rather than a new subclause to the Reservations model subclause.

Related Documents

02-483r0 (November 6, 2002) Further Work on Persistent Reservations. First problem statement following on SPC-2 work..
02-483r1 (May 5, 2003) Two Persistent Reservations problems - updates & decision required. Details of current usage and suggested new features and their usage explored in detail.
03-231r0 (July 2, 2003) Two Persistent Reservations problems - latest information. Feedback on earlier approaches and results from latest testing & investigation.
spc3r15 – SCSI Primary Commands – 3 revision 15

Overview

Reserves and Releases are presently used extensively in situations where sequential-access devices are employed. However because there is no way of checking for the existence of a reservation, multiple reserves and releases are known to be issued against the same device by applications, device drivers and other entities that coexist in a computer system. When combined with the definitions contained in SPC-2, this situation makes the deployment of persistent reservations very problematic, because it would require the application, platform and other entities to be upgraded simultaneously, and that is unacceptable for a number of both technical and business reasons. A number of approaches to relieving this situation have been investigated during the last year, all of which have provided some scheme for allowing mixed usage of reservations and persistent reservations in accessing a storage device. This proposal treats reserve and release as NOPs if they are generated by a computer system that already has access to the device through an existing persistent peservation. In all other cases, the behavior is as defined in SPC-2.

Proposal

This document proposes a modification to the method of handling RESERVE and RELEASE commands in the presence of a Persistent Reservation that was defined in SPC-2. The modification is proposed for inclusion in SPC-3 and is NOT backwards compatible. This proposal is made to address the need to be able to mix usage of reserve/release and persistent reservations when dealing with tapes as identified in 02-483r1 and further addressed in 03-231r0. An addition to the REPORT CAPBILITIES service action is proposed to indicate that this modified reservation handling is supported. The modification is not selectable.

The proposal consists of two parts:

1) Add three footnotes and two rows to Table 31.
2) Add a bit to the PERSISTENT RESERVE IN parameter data for the REPORT CAPABILITIES service action in subclause 6.11.1

Suggested Changes

5.6.1 Reservations overview

Reservations may be used to allow a device server to execute commands from a selected set of I_T nexuses (i.e., combinations of initiator ports accessing target ports) and reject commands from I_T nexuses outside the selected set. The device server uniquely identifies I_T nexuses using protocol specific mechanisms.

Application clients may add or remove I_T nexuses 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 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; and
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). However, any restrictions based on the type of reservation are independent of the scope of the reservation.

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 31. If any element is reserved within a logical unit, that logical unit shall be considered reserved for the commands listed in table 31 and the allowed/conflict information in table 31 shall apply.

In table 31 and table 32 the following key words are used:

allowed: Commands received from I_T nexuses not holding the reservation or from I_T nexuses not registered when a registrants only or all registrants type persistent reservation is present should complete normally.

conflict: Commands received from I_T nexuses not holding the reservation or from I_T nexuses not registered when a registrants only or all registrants type persistent reservation is present shall not be performed and the device server shall terminate the command with a RESERVATION CONFLICT status.

Commands from I_T nexuses holding a reservation should complete normally. The behavior of commands from registered I_T nexuses when a registrants only or all registrants persistent reservation is present is specified in table 31 and table 32.

An unlinked command shall be checked for reservation conflicts before the task containing that command enters the enabled task state. The reservation state as it exists when the first command in a group of linked commands enters the enabled task state shall be used in checking for reservation conflicts for all the commands in the task. Once a task has entered the enabled task state, the command or commands comprising that task shall not be terminated with a RESERVATION CONFLICT due to a subsequent reservation. Any command in a group of linked commands that changes the reservation state shall be the last command in the group.

For each command, this standard or a related command standard (see 3.1.17) defines the conditions that result in RESERVATION CONFLICT. Command standards define the conditions either in the device model (preferred) or in the descriptions each specific command.

Table 31 — SPC commands that are allowed in the presence of various reservations (part 1 of 2)

Command	Addressed LU has this type of persistent reservation held by another I_T nexus				
	From any I_T nexus		From registered I_T nexus (RR all types)	From I_T nexus not registered	
	Write Excl	Excl Access		Write Excl RR	Excl Access – RR
ACCESS CONTROL IN	Allowed	Allowed	Allowed	Allowed	Allowed
ACCESS CONTROL OUT	Allowed	Allowed	Allowed	Allowed	Allowed
CHANGE ALIASES	Conflict	Conflict	Allowed	Conflict	Conflict
EXTENDED COPY	Conflict	Conflict	Allowed	Conflict	Conflict
INQUIRY	Allowed	Allowed	Allowed	Allowed	Allowed
LOG SELECT	Conflict	Conflict	Allowed	Conflict	Conflict
LOG SENSE	Allowed	Allowed	Allowed	Allowed	Allowed
MODE SELECT(6)/ MODE SELECT(10)	Conflict	Conflict	Allowed	Conflict	Conflict
MODE SENSE(6)/ MODE SENSE(10)	Conflict	Conflict	Allowed	Conflict	Conflict
PERSISTENT RESERVE IN	Allowed	Allowed	Allowed	Allowed	Allowed
PERSISTENT RESERVE OUT	see table 32				
PREVENT ALLOW MEDIUM REMOVAL (Prevent=0)	Allowed	Allowed	Allowed	Allowed	Allowed
PREVENT ALLOW MEDIUM REMOVAL (Prevent<>0)	Conflict	Conflict	Allowed	Conflict	Conflict
READ ATTRIBUTE	Conflict	Conflict	Allowed	Conflict	Conflict
READ BUFFER	Conflict	Conflict	Allowed	Conflict	Conflict

Key: **LU**=Logical Unit, **Excl**=Exclusive, **RR**=Registrants Only or All Registrants, **<>** Not Equal

[a] **When a RELEASE command is received from an I_T nexus which holds a persistent reservation or is registered when a registrants only or all registrants type persistent reservation is present, the command shall be allowed, but the Persistent Reservation shall not be released.**

[b] **When a RESERVE command is received from an I_T nexus that holds a persistent reservation or is registered when a registrants only or all registrants type persistent reservation is present, the command shall be allowed but no Reservation shall established and the persistent reservation shall not be changed.**

[c] **When a RESERVE or RELEASE command is received from an I_T nexus that is not a reservation holder, or from an I_T nexus not registered when a registrants only or all registrants type persistent reservation is present and a persistent reservation exists, or from a registered I_T nexus when no persistent reservation exists, then the command shall not be performed and the device server shall terminate the command with a RESERVATION CONFLICT status.**

Table 31 — SPC commands that are allowed in the presence of various reservations (part 2 of 2)

Command	Addressed LU has this type of persistent reservation held by another I_T nexus				
	From any I_T nexus		From registered I_T nexus (RR all types)	From I_T nexus not registered	
	Write Excl	Excl Access		Write Excl RR	Excl Access – RR
READ MEDIA SERIAL NUMBER	Allowed	Allowed	Allowed	Allowed	Allowed
RECEIVE COPY RESULTS	Conflict	Conflict	Allowed	Conflict	Conflict
RECEIVE DIAGNOSTIC RESULTS	Conflict	Conflict	Allowed	Conflict	Conflict
RELEASE	As defined in SPC-2 [ac]				
REPORT ALIASES	Allowed	Allowed	Allowed	Allowed	Allowed
REPORT DEVICE IDENTIFIER	Allowed	Allowed	Allowed	Allowed	Allowed
REPORT LUNS	Allowed	Allowed	Allowed	Allowed	Allowed
REPORT SUPPORTED OPERATION CODES	Conflict	Conflict	Allowed	Conflict	Conflict
REPORT SUPPORTED TASK MANAGEMENT FUNCTIONS	Conflict	Conflict	Allowed	Conflict	Conflict
REPORT TARGET PORT GROUPS	Allowed	Allowed	Allowed	Allowed	Allowed
REQUEST SENSE	Allowed	Allowed	Allowed	Allowed	Allowed
RESERVE	As defined in SPC-2 [bc]				
SEND DIAGNOSTICS	Conflict	Conflict	Allowed	Conflict	Conflict
SET DEVICE IDENTIFIER	Conflict	Conflict	Allowed	Conflict	Conflict
SET TARGET PORT GROUPS	Conflict	Conflict	Allowed	Conflict	Conflict
TEST UNIT READY	Allowed	Allowed	Allowed	Allowed	Allowed
WRITE ATTRIBUTE	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE BUFFER	Conflict	Conflict	Allowed	Conflict	Conflict

Key: **LU**=Logical Unit, **Excl**=Exclusive, **RR**=Registrants Only or All Registrants, **<>** Not Equal

[a] When a RELEASE command is received from an I_T nexus which holds a persistent reservation or is registered when a registrants only or all registrants type persistent reservation is present, the command shall be allowed, but the Persistent Reservation shall not be released.

[b] When a RESERVE command is received from an I_T nexus that holds a persistent reservation or is registered when a registrants only or all registrants type persistent reservation is present, the command shall be allowed but no Reservation shall established and the persistent reservation shall not be changed.

[c] When a RESERVE or RELEASE command is received from an I_T nexus that is not a reservation holder, or from an I_T nexus not registered when a registrants only or all registrants type persistent reservation is present and a persistent reservation exists, or from a registered I_T nexus when no persistent reservation exists, then the command shall not be performed and the device server shall terminate the command with a RESERVATION CONFLICT status.

Table 32 — PERSISTENT RESERVE OUT service actions that are allowed in the presence of various reservations

Command Service Action	Addressed LU has a persistent reservation held by another I_T nexus	
	Command is from a registered I_T nexus	Command is from a not registered I_T nexus
CLEAR	Allowed	Conflict
PREEMPT	Allowed	Conflict
PREEMPT & ABORT	Allowed	Conflict
REGISTER	Allowed	Allowed
REGISTER AND IGNORE EXISTING KEY	Allowed	Allowed
RELEASE	Allowed [a]	Conflict
RESERVE	Conflict	Conflict
Key: **LU**=Logical Unit		
[a] The reservation is not released (see 5.6.2.7.2).		

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 of the tasks in the task set 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. Any persistent reserve service action shall be performed as a single indivisible event.

Multiple persistent reserve service actions may be present in the task set at the same time. The order of execution of such service actions is defined by the tagged queuing restrictions, if any, but each is executed as a single indivisible command without any interleaving of actions that may be required by other reservation commands.

6.11. PERSISTENT RESERVE IN command

6.11.1 PERSISTENT RESERVE IN command introduction

The PERSISTENT RESERVE IN command (see table 96) is used to obtain information about persistent reservations and reservation keys (i.e., registrations) that are active within a device server. This command is used in conjunction with the PERSISTENT RESERVE OUT command (see 6.12).

The PERSISTENT RESERVE IN parameter data includes a field that indicates the number of parameter data bytes available to be returned. The ALLOCATION LENGTH field in the CDB indicates how much space has been allocated for the returned parameter list. An allocation length that is not sufficient to contain the entire parameter list shall not be considered an error. If the complete list is required, the application client should send a new PERSISTENT RESERVE IN command with allocation length large enough to contain the entire list.

Table 96 — 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							

6.11.2 PERSISTENT RESERVE IN service actions

6.11.2.1 Summary of PERSISTENT RESERVE IN service actions

The service action codes for the PERSISTENT RESERVE IN command are defined in table 97.

Table 97 — PERSISTENT RESERVE IN service action codes

Code	Name	Description
00h	READ KEYS	Reads all registered reservation keys (i.e., registrations)
01h	READ RESERVATION	Reads the current persistent reservations
02h	REPORT CAPABILITIES	Returns capability information
03h - 1Fh	Reserved	Reserved

6.11.2.2 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 I_T nexus' reservation key. If multiple I_T nexuses 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 5.6.2.3.2.

6.11.2.3 READ RESERVATIONS

The READ RESERVATIONS service action requests that the device server return a parameter list containing a header and the persistent reservations, if any, that are present in the device server. Multiple persistent reservations may be returned only if element persistent reservations are present.

For more information on READ RESERVATION see 5.6.2.3.3.

6.11.2.4 REPORT CAPABILITIES

The REPORT CAPABILITIES service action requests that the device server return information on persistent reservation features.

6.11.3 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 98.

Table 98 — PERSISTENT RESERVE IN parameter data for READ KEYS

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				PRGENERATION				(LSB)
4	(MSB)							
7				ADDITIONAL LENGTH (n-7)				(LSB)
	Reservation key list							
8	(MSB)							
15				First reservation key				(LSB)
				. . .				
n-7	(MSB)							
n				Last reservation key				(LSB)

The Persistent Reservations Generation (PRGENERATION) field shall contain a 32-bit counter maintained by the device server that shall be incremented every time a PERSISTENT RESERVE OUT command requests a REGISTER, a REGISTER AND IGNORE EXISTING KEY, 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 terminated due to an error or reservation conflict. Regardless of the APTPL bit value the PRgeneration value shall be set to zero by a power on.

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 first portion of the list (byte 0 to the 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 I_T nexuses that have registered with the device server through all combinations of initiator ports and target ports.

6.11.4 PERSISTENT RESERVE IN parameter data for READ RESERVATION

6.11.4.1 Format of 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 99.

The PRGENERATION field shall be as defined for the PERSISTENT RESERVE IN READ KEYS parameter data (see 6.11.3).

Table 99 — PERSISTENT RESERVE IN parameter data for READ RESERVATION

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				PRGENERATION				(LSB)
4	(MSB)							
7				ADDITIONAL LENGTH (n-7)				(LSB)
8								
n				Reservation descriptor(s) (see table 100)				

The ADDITIONAL LENGTH field contains a count of the number of bytes to follow in 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 first portion of the list (byte 0 to the 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 of 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 100. 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 100 — 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								
15				Obsolete				

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 READ RESERVATION service action. The reservation descriptor for each persistent reservation shall contain the RESERVATION KEY under which the persistent reservation is held (see 5.6.2.6). 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 6.11.4.2 and 6.11.4.3).

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 READ RESERVATION service action for the LU_SCOPE persistent reservation that is held, if any, and each ELEMENT_SCOPE 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 reser-

vation as present in the PERSISTENT RESERVE OUT command that created the persistent reservation shall be returned (see 6.11.4.2 and 6.11.4.3).

If the SCOPE is an ELEMENT_SCOPE persistent reservation, the SCOPE-SPECIFIC ADDRESS field shall contain the element address, zero filled in the most significant bits to fit the field. If the SCOPE is a LU_SCOPE persistent reservation, the SCOPE-SPECIFIC ADDRESS field shall be set to zero. The obsolete field in Bytes 14 and 15 was defined in a previous standard.

6.11.4.2 Persistent reservations Scope

6.11.4.2.1 Summary of 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 101.

Table 101 — Persistent reservation scope codes

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

6.11.4.2.2 Logical unit scope

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

6.11.4.2.3 Element scope

A SCOPE field value of ELEMENT_SCOPE 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 SMC-2 standard. The ELEMENT_SCOPE scope is optional for all device servers that implement PERSISTENT RESERVE OUT.

6.11.4.3 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 102 defines the characteristics of the different type values. For each persistent reservation type, table 102 lists code value and describes the required device server support. In table 102, the description of required device server support is divided into three paragraphs:

1) A definition of the required handling for read operations;
2) A definition of the required handling for write operations; and

3) A definition of the persistent reservation holder (see 5.6.2.6).

Table 102 — Persistent reservation type codes (part 1 of 2)

Code	Name	Description
0h		Obsolete
1h	Write Exclusive	**Reads Shared**: Any application client on any I_T nexus may initiate tasks that request transfers from the storage medium or cache of the logical unit to the I_T nexus. **Writes Exclusive**: Any task from any I_T nexus other than the I_T nexus holding the persistent reservation that requests a transfer from the I_T nexus to the storage medium or cache of the logical unit shall be terminated with RESERVATION CONFLICT status. **Persistent Reservation Holder:** The I_T nexus that delivered the PERSISTENT RESERVE OUT command with RESERVE, PREEMPT, or PREEMPT AND ABORT service action as identified by its registered reservation key.
2h		Obsolete
3h	Exclusive Access	**Reads Exclusive**: Any task from any I_T nexus other than the I_T nexus holding the persistent reservation that requests a transfer from the storage medium or cache of the logical unit to the I_T nexus shall be terminated with RESERVATION CONFLICT status. **Writes Exclusive**: Any task from any I_T nexus other than the I_T nexus holding the persistent reservation that requests a transfer from the I_T nexus to the storage medium or cache of the logical unit shall be terminated with RESERVATION CONFLICT status. **Persistent Reservation Holder:** The I_T nexus that delivered the PERSISTENT RESERVE OUT command with RESERVE, PREEMPT, or PREEMPT AND ABORT service action as identified by its registered reservation key.
4h		Obsolete
5h	Write Exclusive – Registrants Only	**Reads Shared**: Any application client on any I_T nexus may initiate tasks that request transfers from the storage medium or cache of the logical unit to the I_T nexus. **Writes Exclusive:** A task that requests a transfer to the storage medium or cache of the logical unit from a initiator port that is not associated with a registered I_T nexus shall be terminated with RESERVATION CONFLICT status. **Persistent Reservation Holder:** The I_T nexus that delivered the PERSISTENT RESERVE OUT command with RESERVE, PREEMPT, or PREEMPT AND ABORT service action as identified by its registered reservation key.
6h	Exclusive Access – Registrants Only	**Reads Exclusive:** A task that requests a transfer from the storage medium or cache of the logical unit to a initiator port that is not associated with a registered I_T nexus shall be terminated with RESERVATION CONFLICT status. **Writes Exclusive:** A task that requests a transfer to the storage medium or cache of the logical unit from an initiator port that is not currently registered with the device server shall be terminated with RESERVATION CONFLICT status. **Persistent Reservation Holder:** The I_T nexus that delivered the PERSISTENT RESERVE OUT command with RESERVE, PREEMPT, or PREEMPT AND ABORT service action as identified by its registered reservation key.

Table 102 — Persistent reservation type codes (part 2 of 2)

Code	Name	Description
7h	Write Exclusive – All Registrants	**Reads Shared**: Any application client on any I_T nexus may initiate tasks that request transfers from the storage medium or cache of the logical unit to the I_T nexus. **Writes Exclusive:** A task that requests a transfer to the storage medium or cache of the logical unit from a initiator port that is not associated with a registered I_T nexus shall be terminated with RESERVATION CONFLICT status. **Persistent Reservation Holder:** Any registered I_T nexus as identified by a zero reservation key value.
8h	Exclusive Access – All Registrants	**Reads Exclusive:** A task that requests a transfer from the storage medium or cache of the logical unit to a initiator port that is not associated with a registered I_T nexus shall be terminated with RESERVATION CONFLICT status. **Writes Exclusive:** A task that requests a transfer to the storage medium or cache of the logical unit from a initiator port that is not associated with a registered I_T nexus shall be terminated with RESERVATION CONFLICT status. **Persistent Reservation Holder:** Any registered I_T nexus as identified by a zero reservation key value.
9h - Fh	Reserved	

6.11.5 PERSISTENT RESERVE IN parameter data for REPORT CAPABILITIES

The format for the parameter data provided in response to a PERSISTENT RESERVE IN command with the REPORT CAPABILITIES service action is shown in table 103.

Table 103 — PERSISTENT RESERVE IN parameter data for REPORT CAPABILITIES

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1		LENGTH (0008h)						(LSB)
2	Reserved			MRH_C	SIP_C	ATP_C	ES_C	PTPL_C
3	TMV	Reserved						PTPL_A
4		PERSISTENT RESERVATION TYPE MASK						
5								
6		Reserved						
7								

The LENGTH field indicates the length in bytes of the parameter data. If the ALLOCATION LENGTH field in the CDB is too small to transfer all of the parameter data, the length shall not be adjusted to reflect the truncation.

A MRH_C (Modified Reservation Handling Capable) bit set to one indicates that the device server supports the modified reservation handling scheme (see Table 31). An MRH_C bit of zero indicates that RESERVE and RELEASE commands are as defined in SPC-2.

An SIP_C (Specify Initiator Ports Capable) bit set to one indicates that the device server supports the SPEC_I_PT bit in the PERSISTENT RESERVE OUT command parameter data (see 6.12.3). An SIP_C bit set to zero indicates that

the device server does not support the SPEC_I_PT bit in the PERSISTENT RESERVE OUT command parameter data.

An ATP_C (All Target Ports Capable) bit set to one indicates that the device server supports the ALL_TG_PT bit in the PERSISTENT RESERVE OUT command parameter data. An ATP_C bit set to zero indicates that the device server does not support the ALL_TG_PT bit in the PERSISTENT RESERVE OUT command parameter data.

An PTPL_C (Persist Through Power Loss Capable) bit set to one indicates that the device server supports the persist through power loss capability (see 5.6.2.2) for persistent reservations and the APTPL bit in the PERSISTENT RESERVE OUT command parameter data. An PTPL_C bit set to zero indicates that the device server does not support the persist through power loss capability.

An ES_C (Element Scope Capable) bit set to one indicates that the device server supports a SCOPE value of ELEMENT_SCOPE (see 6.11.4.2) in PERSISTENT RESERVE OUT commands (see 6.12). An ES_C bit set to zero indicates that the device server does not support a SCOPE value of ELEMENT_SCOPE in PERSISTENT RESERVE OUT commands.

A TMV (Type Mask Valid) bit set to one indicates the PERSISTENT RESERVATION TYPE MASK field contains a bit map indicating which persistent reservation types are supported by the device server. A TMV bit set to zero indicates the PERSISTENT RESERVATION TYPE MASK field shall be ignored.

A PTPL_A (Persist Through Power Loss Activated) bit set to one indicates that persist through power loss capability (see 5.6.2.2) is activated because the most recent successfully completed PERSISTENT RESERVE OUT command with REGISTER or REGISTER AND IGNORE EXISTING KEY service action had the APTPL bit set to one in the parameter data. A PTPL_A bit set to zero indicates that the persist through power loss capability is not activated.

The PERSISTENT RESERVATION TYPE MASK field (see table 104) contains a bit map that indicates the persistent reservation types that are supported by the device server.

Table 104 — Persistent Reservation Type Mask format

Bit Byte	7	6	5	4	3	2	1	0
4	WR_EX_AR	EX_AC_RO	WR_EX_RO	Reserved	EX_AC	Reserved	WR_EX	Reserved
5	Reserved							EX_AC_AR

A WR_EX_AR (Write Exclusive – All Registrants) bit set to one indicates that the device server supports the Write Exclusive – All Registrants persistent reservation type. An WR_EX_AR bit set to zero indicates that the device server does not support the Write Exclusive – All Registrants persistent reservation type.

An EX_AC_RO (Exclusive Access – Registrants Only) bit set to one indicates that the device server supports the Exclusive Access – Registrants Only persistent reservation type. An EX_AC_RO bit set to zero indicates that the device server does not support the Exclusive Access – Registrants Only persistent reservation type.

A WR_EX_RO (Write Exclusive – Registrants Only) bit set to one indicates that the device server supports the Write Exclusive – Registrants Only persistent reservation type. An WR_EX_RO bit set to zero indicates that the device server does not support the Write Exclusive – Registrants Only persistent reservation type.

An EX_AC (Exclusive Access) bit set to one indicates that the device server supports the Exclusive Access persistent reservation type. An EX_AC bit set to zero indicates that the device server does not support the Exclusive Access persistent reservation type.

An WR_EX (Write Exclusive) bit set to one indicates that the device server supports the Write Exclusive persistent reservation type. An WR_EX bit set to zero indicates that the device server does not support the Write Exclusive persistent reservation type.

An EX_AC_AR (Exclusive Access – All Registrants) bit set to one indicates that the device server supports the Exclusive Access – All Registrants persistent reservation type. An EX_AC_AR bit set to zero indicates that the device server does not support the Exclusive Access – All Registrants persistent reservation type.