To: T10 Technical Committee
From: Rob Elliott, Compaq Computer Corporation (Robert.Elliott@compaq.com)
Date: 3 January 2002
Subject: T10/01-099r5 SPC-3 Letting persistent reservations ignore target ports

Revision History

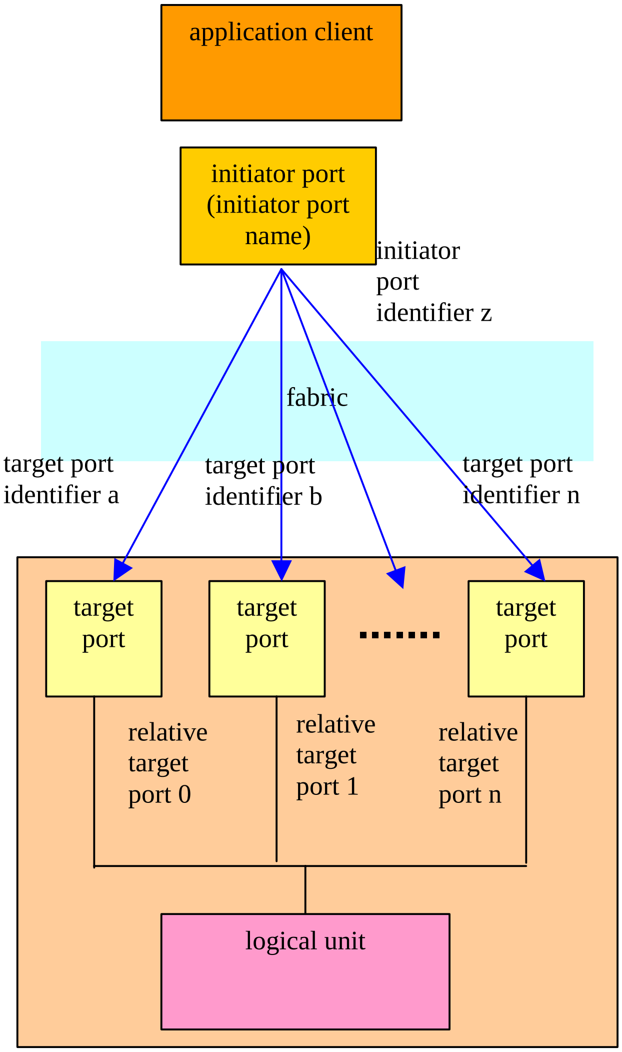
Revision 0 (5 March 2001) first revision. Not complete, but as presented in CAP working group.
Revision 1 (17 April 2001). Removed mode page option; use a bit in the CDB. Removed target port group relationship. Add PR IN report capability.
Revision 2 (26 June 2001). New approach: let target port be ignored if the initiator port name is world wide unique.
Revision 3 (27 July 2001). Incorporated input from July CAP WG. Added set capability to preserve forward/backwards compatibility. Added persist through power loss requirement. Added length field to report capabilities parameter data.
Revision 4 (18 October 2001): Incorporated input from September CAP WG. Restored the ignore target port bit in every REGISTER parameter list.
Revision 5 (3 January 2002): Incorporated input from November CAP WG. Made the bit “all target ports” rather than “ignore target ports” with macro-like behavior.

Related Documents

T10/spc3r02 – SCSI Primary Commands 3 revision 2 (by Ralph Weber)
T10/00-182r0 - SAM-2 device and port names (by Jim Hafner)

Overview

SPC-2 requires that logical units remember the target port through which a reservation was made in addition to the initiator port that originated the reservation. An application client wishing to make a reservation needs to run the PERSISTENT RESERVE command through each target port that can route commands to the logical unit.



In many cases, the logical unit doesn’t care which target port the reservation came through. It just wants to distinguish between initiators. Requiring reservation commands for each port burdens the application client with issuing extra commands.

The reason for this rule is that the target ports may be in different SCSI domains, so they cannot rely on the initiator port identifier alone to truly identify the initiator. If both domains were on parallel SCSI busses, for example, both initiators would likely appear as ID 7.

On many fabrics, however, the initiator port is identified not only by an initiator port identifier (a volatile address) but also an initiator port name (non-volatile, fixed to the port). The initiator port name, if supported by a protocol, is required to be world wide unique. When this is true, the worry about different SCSI domains disappears. If the initiator port name and protocol are the same, the initiator ports must be the same.

This proposal clarifies when the logical unit uses the initiator port identifier and initiator port name in reservations (replacing the current wording about a “worldwide identifier”) and adds a option to apply a registration to all target ports. It adds a PR IN service action to report support for this feature (and some other existing PR features).

Suggested Changes to SPC-3

5.5.1 Reservations overview

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

’’’

5.5.3.1 Overview of the Persistent Reservations management method

…

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 shall 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 an initiator on a specific target 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. These rules provide the ability for an application client to preempt multiple initiators with a single PERSISTENT RESERVE OUT command, but they do not provide the ability for the application client to uniquely identify the initiators using the PERSISTENT RESERVE commands.

5.5.3.2 Preserving persistent reservations

The application client may request activation of the persist through power loss device server capability to preserve the persistent reservation and registration keys across power cycles by setting the APTPL bit to one in PERSISTENT RESERVE OUT parameter data sent with a REGISTER~~,~~ or ~~a~~ REGISTER AND IGNORE EXISTING KEY service action.

After the application client enables the persist through power loss capability the device server shall preserve all current and future registrations and persistent reservations associated with the logical unit to which the REGISTER or the REGISTER AND IGNORE EXISTING KEY service action was addressed until an application client disables the persist through power loss capability. The APTPL value from the most recent successfully completed REGISTER or REGISTER AND IGNORE EXISTING KEY service action from any application client shall determine the logical unit’s behavior in the event of a power loss.

The device server shall preserve the following information for each registration across any reset, and if the persist through power loss capability is enabled, across any power cycle:
a) ~~Initiator identifier~~On protocols where initiator port names are required, the initiator port name. On protocols where initiator port names are not required, the initiator port identifier;
b) reservation key; and

~~c) when supported by the protocol, the initiator port's world wide identification~~indication of which target port was used to make the registration.

The device server shall preserve the following reservation information across any reset, and if the persist through power loss capability is enabled, across any power cycle:

a) ~~Initiator identifier;~~On protocols where initiator port names are required, the initiator port name. On protocols where initiator port names are not required, the initiator port identifier;
b) reservation key;
c) scope;
d) type; and
e) ~~when supported by the protocol, the initiator port's world wide identification.~~indication of which target port was used to make the reservation.

~~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 of the initiator port having the same world wide identification.~~

…

5.5.3.4 Registering

…

In response to a PERSISTENT RESERVE OUT with a REGISTER or a REGISTER AND IGNORE EXISTING KEY 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~~ target port being used for the registration and either the initiator port name on protocols where port names are required or the initiator port identifier on protocols where port names are not required;
e) register the reservation key without changing any ~~a~~ persistent reservation that may exist; and
f) retain the reservation key and associated information.

5.6 Multiple port ~~and multiple initiator~~ behavior

SAM-2 specifies the behavior of logical units being accessed by more than one initiator. Additional ~~service delivery~~target ports provide alternate ~~service delivery~~ paths through which the device server may be reached ~~and may also provide connectivity for additional initiators~~. An alternate path may be used to improve the availability of devices in the presence of certain types of failures and to improve the performance of devices whose other paths may be busy.

If a SCSI target device has more than one ~~service delivery~~target port, the arbitration and connection management among the ~~service delivery~~target ports is vendor specific. If one ~~service delivery~~target port is being used by an initiator, accesses attempted through other ~~service delivery~~target port(s) may:

a) receive a status of BUSY; or
b) be accepted as if the other ~~service delivery~~target port(s) were not in use.

The device server shall indicate the presence of multiple target ports by setting the MULTIP bit to ~~1~~ one in its standard INQUIRY data.

For the purposes of handling reservations, other initiators are defined as all initiators ~~on the same service delivery port except the initiator holding the reservation and all initiators on all other service delivery ports.~~ accessing the logical unit through the same target port except the initiator holding the reservation and all initiators accessing the logical unit through all other target ports.

Only the following operations allow an initiator to interact with the tasks of ~~an~~other initiators, regardless of the ~~service delivery~~target port:

a) the PERSISTENT RESERVE OUT with PREEMPT service action preempts persistent reservations for other initiators (see 5.5.3.6.3);

b) the PERSISTENT RESERVE OUT with PREEMPT AND ABORT service action preempts persistent reservations and aborts all tasks for other initiators (see 5.5.3.6.4);

[Editor's note: tasks are not preempted, they are aborted]

c) the PERSISTENT RESERVE OUT with CLEAR service action releases persistent reservations and removes reservation keys for all initiators (see 5.5.3.6.5);

d) the TARGET RESET task management function releases reservations established by the reserve/release method and removes all tasks for all initiators for all logical units in the target ~~and for all initiators~~ (see SAM-2). Persistent reservations remain unmodified;

e) the LOGICAL UNIT RESET task management function releases reservations established by the reserve/release method and removes all tasks for all initiators for the addressed logical unit and any logical units issuing from it in a hierarchical addressing structure (see SAM-2). Persistent reservations remain unmodified; and

f) the CLEAR TASK SET task management function removes all tasks for all initiators for the selected logical unit ~~for all initiators~~. Most other logical unit states remain unmodified, including MODE SELECT parameters, reservations, and ACA (see SAM-2).

[Editor's note: just making d) e) and f) use the same order for "for all initiators"]

7.10.2 PERSISTENT RESERVE IN service actions

7.10.2.1 Summary of PERSISTENT RESERVE IN service actions

The service action codes for the PERSISTENT RESERVE IN command are defined in table 67.

Table 67 — PERSISTENT RESERVE IN service action codes

Code	Name	Description
00h	READ KEYS	Reads all registered Reservation Keys
01h	READ RESERVATION	Reads the current persistent reservations
02h	REPORT CAPABILITIES	Returns capability information
~~02h~~ 03h – 1Fh	Reserved	Reserved

7.10.2.x Report Capabilities

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

Table 73. REPORT CAPABILITIES parameter data

Byte\Bit	7	6	5	4	3	2	1	0
0	(MSB)	LENGTH (0008h)						
1								(LSB)
2	Reserved					ALL TARGET PORTS CAPABLE	ELEMENT SCOPE CAPABLE	APTPL CAPABLE
3	Reserved						Rsvd	PTPL
4	Reserved							
5	Reserved							
6	Reserved							
7	Reserved							

The LENGTH field specifies 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.

An ALL TARGET PORTS CAPABLE bit of one indicates that the device server supports the ALL TARGET PORTS bit in the PERSISTENT RESERVE OUT command. An ALL TARGET PORTS CAPABLE bit of zero indicates that the device server does not support the ALL TARGET PORTS bit in the PERSISTENT RESERVE OUT command.

An Activate Persist Through Power Loss Capable (APTPL CAPABLE) bit of one indicates that the device server supports the APTPL bit in the PERSISTENT RESERVE OUT command. An APTPL CAPABLE bit of zero indicates that the device server does not support the APTPL bit in the PERSISTENT RESERVE OUT command.

An ELEMENT SCOPE CAPABLE bit of one indicates that the device server supports a SCOPE value of ELEMENT_SCOPE in the persistent reservation commands. An ELEMENT SCOPE CAPABLE bit of zero indicates that the device server does not support a SCOPE value of ELEMENT_SCOPE in the persistent reservation commands.

A Persist Through Power Loss (PTPL) bit of one indicates that the most recent successfully completed REGISTER or REGISTER AND IGNORE EXISTING KEY service action had its APTPL bit set to one (see 7.11.3). A PTPL bit of zero indicates that the most recent successfully completed REGISTER or REGISTER AND IGNORE EXISTING KEY service action had its APTPL bit set to zero (see 7.11.3).

7.11 PERSISTENT RESERVE OUT

…

7.11.2 PERSISTENT RESERVE OUT Service Actions

When processing the PERSISTENT RESERVE OUT service actions, the device server shall increment the generation value as specified in 7.10.3.

The PERSISTENT RESERVE OUT command service actions are defined in table 74.

Table 74 — PERSISTENT RESERVE OUT service action codes

Code	Name	Description	GENERATION field incremented (see 7.10.3)
00h	REGISTER	<as is>	Yes
01h	RESERVE	<as is>	No
02h	RELEASE	<as is>	No
03h	CLEAR	<as is>	Yes
04h	PREEMPT	<as is>	Yes
05h	PREEMPT AND ABORT	<as is>	Yes
06h	REGISTER AND IGNORE EXISTING KEY	<as is>	Yes
07h – 1Fh	Reserved	Reserved	

The parameter list values for each service action are specified in 7.11.3.

7.11.3 PERSISTENT RESERVE OUT parameter list

The parameter list required to perform the PERSISTENT RESERVE OUT command is defined in table 75. All fields shall be sent on all PERSISTENT RESERVE OUT commands, even if the field is not required for the specified service action and scope values.

Table 75 — PERSISTENT RESERVE OUT parameter list

Byte\Bit	7	6	5	4	3	2	1	0
0	RESERVATION KEY							
7								
8	SERVICE ACTION RESERVATION KEY							
15								
16	SCOPE-SPECIFIC ADDRESS							
19								
20	Rsvd	Rsvd	Rsvd	Rsvd	Rsvd	~~Rsvd~~ALL TARGET PORTS	Rsvd	APTL
21		Reserved						
22	Obsolete							
23								

The obsolete field in Bytes 22 and 23 was defined in a previous standard for use with an obsolete scope (see table 71). If the obsolete scope is not supported Bytes 22 and 23 should be zero.

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 contents of the RESERVATION KEY field in a PERSISTENT RESERVE OUT command parameter data matches the registered reservation key for the initiator from which the task was received, except for:
a) the REGISTER AND IGNORE EXISTING KEY service action where the RESERVATION KEY field shall be ignored; and
b) the REGISTER service action for an unregistered initiator where the RESERVATION KEY field shall contain zero.

Except as noted above, when 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. Except as noted above, the reservation key of the initiator shall be verified to be correct regardless of the SERVICE ACTION and SCOPE field values.

The SERVICE ACTION RESERVATION KEY field contains information needed for four service actions; the REGISTER, REGISTER AND IGNORE EXISTING KEY, PREEMPT, and PREEMPT AND ABORT service actions. For the REGISTER and REGISTER AND IGNORE EXISTING KEY 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 field is ignored for all other service actions.

If the scope is an ELEMENT_SCOPE reservation, the SCOPE-SPECIFIC ADDRESS field shall contain the element address, zero filled in the most significant bits to fit the field. If the service action is REGISTER, REGISTER AND IGNORE EXISTING KEY, or CLEAR or if the scope is a LU_SCOPE reservation, the SCOPE-SPECIFIC ADDRESS field shall be set to zero.

The Activate Persist Through Power Loss (APTPL) bit ~~shall be~~ is valid only for the REGISTER~~, or~~ and the REGISTER AND IGNORE EXISTING KEY service actions. 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 or a REGISTER AND IGNORE EXISTING KEY service action, the device server shall return a CHECK CONDITION status. The sense key shall be set to ILLEGAL REQUEST and additional sense code 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 5.5.3.4). If the last valid APTPL bit value received by the device server is one, the logical unit shall retain any persistent reservation(s) that may be present and all reservation keys for all initiators even if power is lost and later returned (see 5.5.3.2).

The ALL TARGET PORTS bit is valid only for the REGISTER and REGISTER AND IGNORE EXISTING KEY service actions, and shall be ignored for all other service actions. Support for the ALL TARGET PORTS bit is optional. If the device server receives a REGISTER or REGISTER AND IGNORE EXISTING KEY service action with the ALL TARGET PORTS bit set to one, it shall apply the registration to all target ports. If the device server receives a REGISTER or REGISTER AND IGNORE EXISTING KEY service action with the all target ports bit set to zero, it shall apply the registration only to the target port through which the PERSISTENT RESERVE OUT command was received.

Table 76 summarizes which fields are set by the application client and interpreted by the device server for each service action and scope value. ~~The APTPL bit in the PERSISTENT RESERVE OUT parameter list, specified in the previous paragraph, is not summarized in table 76.~~

Table 76 — PERSISTENT RESERVE OUT service actions and valid parameters

		Parameters					
Service action	**Allowed SCOPE**	**TYPE**	**RESERVATION KEY**	**SERVICE ACTION RESERVATION KEY**	**SCOPE-SPECIFIC ADDRESS**	**ALL TARGET PORTS**	**APTPL**
REGISTER	ignored	~~ignored~~I	~~valid~~V	~~valid~~V	~~ignored~~I	V	V
REGISTER AND IGNORE EXISTING KEY	ignored	~~ignored~~I	~~ignored~~I	~~valid~~V	~~ignored~~I	V	V
RESERVE	LU_SCOPE ~~ELEMENT_SCOPE~~	~~valid~~V ~~valid~~V	~~valid~~V	~~ignored~~I ~~ignored~~I	~~ignored~~I ~~valid~~V ~~(element)~~	I	I
	ELEMENT_SCOPE	V	V	I	V	I	I
RELEASE	LU_SCOPE ~~ELEMENT_SCOPE~~	~~valid~~V ~~valid~~V	~~valid~~V	~~ignored~~I ~~ignored~~I	~~ignored~~I ~~valid~~V ~~(element)~~	I	I
	ELEMENT_SCOPE	V	V	I	V	I	I
CLEAR	ignored	~~ignored~~I	~~valid~~V	~~ignored~~I	~~ignored~~I	I	I
PREEMPT	LU_SCOPE ~~ELEMENT_SCOPE~~	~~valid~~V ~~valid~~V	~~valid~~V	~~valid~~V ~~valid~~V	~~ignored~~I ~~valid~~V ~~(element)~~	I	I
	ELEMENT_SCOPE	V	V	V	V	I	I
PREEMPT & ABORT	LU_SCOPE ~~ELEMENT_SCOPE~~	~~valid~~V ~~valid~~V	~~valid~~V	~~valid~~V ~~valid~~V	~~ignored~~I ~~valid~~V ~~(element)~~	I	I
	ELEMENT_SCOPE	V	V	V	V	I	I

Key: V = valid I =ignored