To: T10 Technical Committee
From: Rob Elliott, Compaq Computer Corporation (Robert.Elliott@compaq.com)
Date: 7 November 2001
Subject: T10/01-318r1 SAM-2 SPC-3 Eliminate SCSI-2 references and describe CA

Revision History

Revision 0 (1 November 2001) first revision
Revision 1 (7 November 2001) included SEND command excerpt and comments from November T10 meeting.

Related Documents

s2-r10l SCSI-2 (Larry Lamers)
sam2r20 SCSI-3 Architecture Model - 2 revision 20 (Ralph Weber)
spc3r01 SCSI-3 Primary Commands - 3 revision 1 (Ralph Weber)
99-207r0 SAM-2 changes for queuing (Ralph Weber) - proposed a) clarifying that only tagged tasks can receive TASK SET FULL, and b) that protocols are allowed to not support untagged tasks. Withdrawn after discussion in July CAP meeting was derailed about how CA handles untagged tasks.
"Where is your Allegiance?" by Ralph Weber and Dal Allen, published in the January 2001 ENDL Letter and posted to IETF IPS reflector on 18 February 2001. Available from http://www.pdl.cmu.edu/mailinglists/ips/mail/msg03427.html.
00-359r7 Unit Attention Issue [Unit Attention Interlock] (Ed Gardner)

Overview

SCSI-2 was published in 1993. In 2001, the second generation of the SCSI-3 Architecture model and the third generation of the SCSI-3 Primary Command set should not need to reference SCSI-2 any more. This creates lots of confusion for users of SCSI, who still refer to SCSI-2 and are thus not aware of many new features.

The key technical feature deferred to SCSI-2 is the definition of contingent allegiance (CA). The debate was whether CA can be cleared with tagged REQUEST SENSE commands, with the consensus to leave in a reference to the vague wording in SCSI-2 rather than copy that vague wording into SAM-2 or agreeing on the behavior. New protocols like SRP do not even support untagged commands, so I think this issue has been solved.

There are a few references to Common Access Method (CAM), which is a SCSI-2 standard, not SCSI-3. It should not be referenced unless the CAM3 project (a SCSI-3 version) is revived.

SCSI-2 holds the only definition of scanner and communications device commands. SPC-3 references SCSI-2 for them in its opcode/mode page/log page tables. Although the codes should never be reused, the reference can be marked obsolete.

This proposal:
1) incorporates CA into SAM-2
2) removes the rule that protocols must support untagged tasks
3) removes references to SCSI-2 (both "SCSI-2" and to the ISO version) CAM, and SCSI-2 SSA.
4) marks scanners and communication devices as obsolete devices

Suggested Changes to SAM-2

1.3 SCSI standards family

…

~~Serial Storage Architecture SCSI-2 Protocol SSA-S2P [ANSI X3.294:1996]~~
Serial Storage Architecture SCSI-3 Protocol SSA-S3P [ANSI NCITS.309:1998]

…

~~Common Access Method:~~
~~SCSI Common Access Method CAM [ISO/IEC 9316-421][ANSI X3.232:1996]~~

The term SCSI is used to refer to the family of standards described in this subclause. ~~The Small Computer System Interface - 2 standard (ISO/IEC 9316:1995-11) and the architecture that it describes are referred to herein as SCSI-2.~~

2.2 Approved references

…

ISO/IEC 60027-2-am2 (1999-01), Letter symbols to be used in electrical technology - Part 2: Telecommunications and electronics (Amendment 2)

~~*ISO/IEC 9316:1995-11,* Small Computer System Interface -2 standard, (SCSI-2) [ANSI X3.270:1996]~~

ISO/IEC 14776-312, SCSI Primary Commands - 2 (SPC-2) [ANSI NCITS.351:~~200x~~2001]

2.3 References under development

At the time of publication, the following referenced standards were still under development. For information on the current status of the document, or regarding availability, contact the relevant standards body or other organization as indicated.

ISO/IEC 14776-313, SCSI Primary Commands - 3 (SPC-3) [T10/1416-D]

*ISO/IEC xyzzy-abc, SCSI Parallel Interface - 4 (*SPI-4) [T10/1365-D]

3.1.2 ACA command: A command performed by a task with the ACA attribute (see 3.3, 3.1.5, ~~and~~ 4.9, and 7.6.4).

3.1.5 auto contingent allegiance (ACA): One of the possible conditions of a task set following the return of a CHECK CONDITION status. See 5.8.1.

3.1.6 blocked task state: The state of a task that is prevented from completing due to an ACA condition.

3.1.19 contingent allegiance (CA): One of the possible conditions of a task set following the return of a CHECK CONDITION status. ~~A detailed definition of CA may be found in SCSI-2.~~ See 5.8.1.

3.1.33 faulted initiator: The initiator to which a CHECK CONDITION status was returned. The faulted initiator condition disappears when the ACA or CA condition resulting from the CHECK CONDITION status is cleared.

3.1.34 faulted task set: A task set that contains a faulting task. The faulted task set condition disappears when the ACA or CA condition resulting from the CHECK CONDITION status is cleared.

3.1.132 task set: A group of tasks within a logical unit, whose interaction is dependent on the task management (queuing), CA, and ACA rules. (See 4.8.)

3.2 Acronyms

…

ACA Auto Contingent Allegiance (see 3.1.5)
AER Asynchronous Event Reporting
CA Contingent Allegiance (see 3.1.19~~13 and SCSI-2~~)
SCSI The architecture defined by the family of standards described in 1.3
~~SCSI-2 The architecture defined by the Small Computer System Interface - 2 standard (ISO/IEC 9316:1995-11)~~
~~SIM SCSI Interface Module (a component of CAM software, see CAM)~~

4.4 The SCSI structural model

The SCSI structural model represents a view of the elements comprising a SCSI I/O system as seen by the application clients interacting with the system. ~~This view is similar to that seen by a CAM device driver interacting with the system through the CAM SIM layer.~~ As shown in figure 7, the fundamental object is the SCSI domain that represents an I/O system. A domain is made up of SCSI devices and a service delivery subsystem that transports commands and data. A SCSI device contains application clients or device servers or both and the infrastructure to support them.

4.8 Logical units

…

(Page 29)

For convenience, task (see 4.9) refers to either a tagged task or an untagged task. The interactions among the tasks in a task set are determined by the rules for task set management specified in clause 7 and the ACA and CA rules specified in 5.8.1 ~~and SCSI-2~~. The number of task sets per logical unit and the boundaries between task sets are governed by the TST field in the Control mode page (see SPC-2).

4.9 Tasks (Page 30)

4.9.1 The task object

The task object represents either a tagged task or an untagged task. The composition of a task includes a definition of the work to be performed by the logical unit in the form of a command or a group of linked commands. A tagged task is composed of a definition of the work to be performed by the logical unit, a tagged task identifier (see 4.9.3), a task attribute (see 7.6), and an I_T_L_Q nexus (see 4.10). An untagged task is composed of a definition of the work to be performed by the logical unit, a untagged task identifier (see 4.9.3), an I_T_L nexus (see 4.10), and implicitly a SIMPLE task attribute (see 7.6). Task identifier (see 4.9.3) refers to either a tagged task identifier or an untagged task identifier.

A tagged task includes a tag (see 4.9.2) in its tagged task identifier that allows many uniquely identified tagged tasks to be present concurrently in a single task set. A tagged task also includes one of the task attributes described in 7.6 that allows the initiator to specify processing relationships between various tagged tasks. An untagged task does not include a tag in any of its component definitions, thus restricting the number of concurrent untagged tasks in a single task set to one per initiator. Also, an untagged task is assumed to have a SIMPLE task attribute, leaving the initiator no control over its relationship to other tasks in the task set.

~~Every SCSI protocol shall support tagged and untagged tasks. Support for tagged tasks by a logical unit is optional.~~

Every SCSI protocol shall support tagged tasks and may support untagged tasks.

If the SCSI protocol upon which a SCSI device operates supports untagged tasks, then the SCSI device is not required to support tagged tasks.

A task identifier that is in use shall be unique as seen by the initiator originating the command and the logical unit to which the command was addressed. A task identifier is in use over the interval bounded by the events specified in 5.5). A task identifier is unique if one or more of its components is unique within the scope specified above. By implication, therefore, an initiator shall not cause the creation of more than one untagged task having identical values for the target identifier and logical unit number. Conversely, an initiator may create more than one task with the same tag value, provided at least one of the remaining task identifier components is unique.

5.2.3 CONTROL **byte**

…

~~The NACA (Normal ACA) bit is used to control the rules for handling an ACA condition caused by the command. The actions to be taken by a logical unit in response to an ACA condition for NACA bit values of one or zero are specified in 5.8.1.1. All logical units shall implement support for the NACA value of zero and may support the NACA value of one. The ability to support a NACA value of one is indicated in standard INQUIRY data (see SPC-2).~~

The NACA (Normal ACA) bit is used to select whether a contingent allegiance (CA) or an auto contingent allegiance (ACA) is established if the command returns with CHECK CONDITION status. An NACA bit of one indicates that an ACA shall be established. An NACA bit of zero indicates that a CA shall be established. The actions for ACA and CA are specified in 5.8.1.1.

All logical units shall support a NACA value of zero (CA) and may support a NACA value of one (ACA). The ability to support a NACA value of one is indicated with the NORMACA bit in the standard INQUIRY data (see SPC-3).

If the NACA bit is set to ~~a value that is not supported~~one but the logical unit does not support ACA, the logical unit shall complete the command with a CHECK CONDITION status with a sense key of ILLEGAL REQUEST and an additional sense code of INVALID FIELD IN CDB~~status of CHECK CONDITION and a sense key of ILLEGAL REQUEST. The rules for handling the resulting ACA condition shall be in accordance with the supported bit value~~ and establish a CA condition.

5.3 Status
5.3.1 Status codes
30h ACA ACTIVE
…
ACA ACTIVE. This status shall be returned when an ACA exists within a task set and an initiator issues a command for that task set when at least one of the following is true:
a) There is a task with the ACA attribute (see 7.6.4) in the task set;
b) The initiator issuing the command did not cause the ACA condition; or
c) The task created to process the command did not have the ACA attribute, and the NACA bit was set to one in the CDB CONTROL byte of the faulting command (see 5.8.1).

The initiator may reissue the command after the ACA condition has been cleared.

5.3.2 Status precedence
If more than one condition applies to a completed task, the report of a BUSY, RESERVATION CONFLICT, ACA ACTIVE or TASK SET FULL status shall take precedence over the return of any other status for that task.

5.6 Aborting tasks
5.6.1 Mechanisms that cause tasks to be aborted
…
The following initiator actions affect only the task(s) created by the initiator that takes the action:
a) Completion of an ABORT TASK task management function directed to the specified task;
b) Completion of an ABORT TASK SET task management function under the conditions specified in 6.3;
c) An ACA or CA condition was cleared and the QERR field was set to 11b in the Control mode page (see SPC-~~2~~3); or
d) An ACA condition was cleared and the task had the ACA attribute ~~(see 6.4)~~.

The following initiator actions affect the task(s) created by the initiator that takes the action and/or task(s) created by other initiators:
a) Completion of a CLEAR TASK SET task management function referencing the task set containing the specified task;
b) An ACA or CA condition was cleared and the QERR field was set to 01b in the Control mode page (see SPC-~~2~~3);
c) Completion of a PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action directed to the initiator that created the task (see SPC-3);
d) A logical unit reset (see 5.8.7); or
e) A hard reset (see 5.8.6).

5.8 Command processing considerations and exception conditions
5.8.1 Auto Contingent Allegiance (ACA) ~~or~~ and Contingent Allegiance (CA)
5.8.1.00 Overview
[Editor's note: An introduction to ACA, CA and autosense was requested in the November CAP meeting. This section does not include any shalls. The SAM-2 editor is free to rewrite this.]

There are two mechanisms for returning sense data when a command completes with a CHECK CONDITION status: autosense and the REQUEST SENSE command. Command acceptance differs under CA and ACA. Table xx provides an overview of how autosense, CA, and ACA interact.

Table xx. Autosense, CA, and ACA interaction.

Protocol supports autosense	Condition	Mechanism to obtain sense data	Mechanism to clear condition	Tasks blocked
no	CA	REQUEST SENSE	any command	no
no	ACA	REQUEST SENSE	any command without ACA task attribute	yes
yes	CA	Autosense	autosense	no
yes	ACA	Autosense	any command without ACA task attribute	yes

On protocols where autosense (see 5.8.4.3) is supported, autosense provides the sense data with a CHECK CONDITION status. If a CA is established, autosense clears the CA. If an ACA is established, command processing is blocked. The initiator may send multiple commands one at a time using the ACA task attribute to handle the ACA condition without clearing it. This also prevents subsequent commands in-flight after the command which caused the ACA from being processed. The initiator sends a REQUEST SENSE command to clear the ACA.

On protocols where autosense is not supported (i.e., SPI-4 with information units disabled), the logical unit enters a CA or an ACA condition after returning a CHECK CONDITION status. The initiator runs a REQUEST SENSE command to retrieve the sense data. If a CA is established, the initiator sends a REQUEST SENSE command as the next command to retrieve the sense data and clear the CA, since the next command clears the CA. If an ACA is established, the initiator may send multiple commands one at a time using the ACA task attribute to handle the ACA condition before sending a REQUEST SENSE command to clear the ACA.

5.8.1.0 Establishing a CA or ACA

~~The ACA (NACA equals one, see 5.2.3) or CA (NACA equals 0) condition shall exist within the task set when the logical unit completes a command by returning a CHECK CONDITION status (see 5.3).~~

When a logical unit completes a command by returning a CHECK CONDITION status, either an ACA or CA condition is created within the task set. If the NACA bit was set to zero in the CONTROL byte of the faulting command, the device server shall create a CA condition. If the NACA bit was set to one in the CONTROL byte of the faulting command, the device server shall create an ACA condition.

After sending the CHECK CONDITION status and returning a service response of TASK COMPLETE, the logical unit shall modify the state of all tasks in the faulted task set as described in clause 7. A task created by the faulted initiator while ~~the~~ an ACA condition is in effect may be entered into the faulted task set under the conditions described ~~below~~in 5.8.1.2aa.

~~As described in 5.8.1.2, the setting of the NACA bit in the CONTROL byte of the faulting command CDB determines the rules that apply to an ACA or CA condition caused by that command.~~

~~If the NACA bit was set to zero in the CONTROL byte of the faulting command, the SCSI-2 CA rules shall apply.~~

~~If the NACA bit was set to one in the CONTROL byte of the faulting command, then a new task created by the faulted initiator while the ACA condition is in effect shall not be entered into the faulted task set unless all of the following conditions are true:~~

~~a) The task has the ACA attribute; and~~

~~b) No other task from the faulted initiator having the ACA attribute is in the task set.~~

~~If the task is from the faulted initiator and any of the conditions listed above are not met, the newly created task shall not be entered into the task set and shall be completed with a status of ACA ACTIVE.~~

~~If a task having the ACA attribute is received and no ACA condition is in effect for the task set or if the NACA bit was set to zero in the CDB for the faulting command (i.e., a CA condition is in effect), then the ACA task shall be completed with a status of CHECK CONDITION. The sense key shall be set to ILLEGAL REQUEST with an additional sense code of INVALID MESSAGE ERROR. Any existing CA condition shall be cleared (see 5.8.1.2) and a new ACA (NACA equals one) or CA (NACA equals zero) condition shall be established.~~

5.8.1.1 ~~Logical Unit response to Auto Contingent Allegiance (ACA) or Contingent Allegiance (CA)~~Handling tasks when neither CA or ACA is in effect

The ACA ~~(NACA equals one, see 5.2.3)~~ or CA ~~(NACA equals zero)~~ condition shall not cross task set boundaries and shall be preserved until it is cleared as described in 5.8.1.2 and 5.8.1.2a. If requested by the application client and supported by the protocol and logical unit, sense data shall be returned as described in 5.8.4.3.

NOTES

~~6 The SCSI-2 CA condition has had an alternate added and the extended contingent allegiance condition has been replaced in SCSI by ACA in conjunction with the NACA bit.~~

7 If the SCSI protocol does not enforce state synchronization as described in 4.6.1, there may be a time delay between the occurrence of the ACA or CA condition and the time at which the initiator becomes aware of the condition.

Table xx describes handling tasks when neither a CA nor an ACA condition is in effect.

Table xx. Handling tasks when neither CA nor ACA condition is in effect

Task attribute of new task	NACA bit in CONTROL byte of CDB of new task	Description
ACA Task	0	Device server shall complete the command with CHECK CONDITION status with sense key of ILLEGAL REQUEST and additional sense code of INVALID MESSAGE ERROR. Device server shall create a CA.
ACA Task	1	Device server shall complete the command with CHECK CONDITION status with sense key of ILLEGAL REQUEST and additional sense code of INVALID MESSAGE ERROR. Device server shall create an ACA.
other	0	If the command fails, the device server shall create a CA.
other	1	If the command fails, the device server shall create an ACA.

5.8.1.2aa Handling tasks from the faulted initiator during CA or ACA

While a CA condition is in effect, processing of all tagged I/O processes from the faulted initiator shall either be aborted or blocked until the CA condition is cleared, based on the TST and QERR fields in the control mode page (see SPC-3).

Table yy describes handling tasks from the faulted initiator when a CA or ACA condition is in effect.

Table yy. Handling tasks from the faulted initiator during CA or ACA

Condition	Task attribute of new task	NACA bit in CONTROL byte of CDB of new task	Description
CA	ACA	0	Device server shall complete the command with CHECK CONDITION status with sense key of ILLEGAL REQUEST and additional sense code of INVALID MESSAGE ERROR. Device server shall clear the CA and establish a new CA.
CA	ACA	1	Device server shall complete the command with CHECK CONDITION status with sense key of ILLEGAL REQUEST and additional sense code of INVALID MESSAGE ERROR. Device server shall clear the CA and establish an ACA.
CA	other	0	Device server shall clear the CA. If the command fails [fails = terminates with CHECK CONDITION status], the device server shall create a CA.
CA	other	1	Device server shall clear the CA. If the command fails, the device server shall create an ACA.
ACA	ACA	0	If a task from the faulted initiator is in the task set with its ACA attribute set, the device server shall return ACA ACTIVE status. Otherwise, it may accept the task. If the command is accepted and fails, the device server shall clear the ACA and create a CA.
ACA	ACA	1	If a task from the faulted initiator is in the task set with its ACA attribute set, the device server shall return ACA ACTIVE status. Otherwise, the device server may accept the task. If the command is accepted and fails, the device server shall clear the ACA and create a new ACA.
ACA	other	0 or 1	Device server shall return ACA ACTIVE status.

5.8.1.2a Handling tasks from other initiators during CA or ACA

The handling of tasks created by initiators other than the faulted initiator depends on the value in the TST field in the Control mode page (see SPC-2).

~~If the TST field contains 000b, tasks created by other initiators while the ACA or CA condition is in effect shall not be entered into the faulted task set, except for a PERSISTENT RESERVE command with a Preempt and Clear service action (see 5.8.1.2). Tasks rejected from the task set due to the presence of an ACA or CA condition shall be completed with a status of ACA ACTIVE (if NACA equals one in the rejected command's CDB CONTROL byte, see 5.2.3) or BUSY (if NACA equals zero).~~

~~If the TST field contains 001b, tasks created by one initiator shall not be rejected based on an ACA or CA condition in effect for another initiator. Only ACA or CA condition for the sending initiator (as well as other task set management considerations described in clause 7) shall affect acceptance into the task set or rejection for a task from that initiator.~~

Table zz describes handling tasks from initiators other than the faulted initiator when a CA or ACA condition is in effect.

Table zz. Handling tasks from other initiators during CA or ACA

Condition	NACA bit in CONTROL byte of CDB of new task	TST field in Control mode page	Description
CA	0 or 1	000b	If the command is PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action, the device server shall accept it and process it. Otherwise, the device server shall return BUSY status.
CA	0 or 1	001b	Device server shall not refuse to accept the task because of the CA.
ACA	0	000b	If the command is PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action, the device server shall accept it and process it. Otherwise, the device server shall return BUSY status.
ACA	1	000b	If the command is PERSISTENT RESERVE OUT command with a PREEMPT AND ABORT service action, the device server shall accept it and process it. Otherwise, the device server shall return ACA ACTIVE status.
ACA	0 or 1	001b	Device server shall not refuse to accept the task because of the ACA.

5.8.1.2b Clearing a CA condition

A CA condition shall only be cleared:

a) as a result of a power on or logical unit reset (see 5.8.7);

b) by an ABORT TASK SET task management function from the faulted initiator;

c) by a CLEAR TASK SET task management function from any initiator if TST is zero;

cc) by a CLEAR TASK SET task management function from the faulted initiator if TST is one;

d) through a PREEMPT AND ABORT service action of a PERSISTENT RESERVE OUT command from another initiator that clears the tasks of the faulted initiator (see SPC-3);

e) upon accepting any subsequent command for the I_T_L nexus; or

f) upon sending sense data by means of the autosense mechanism (see 5.8.4.3).

While a CA is active a logical unit that supports the CLEAR ACA task management function shall ignore all CLEAR ACA requests and shall return a service response of FUNCTION COMPLETE (see 6.4).

5.8.1.2 Clearing an ACA ~~Auto Contingent Allegiance (ACA)~~ condition

~~If the NACA bit is set to zero in the CONTROL byte of the faulting command, then the SCSI-2 rules for clearing CA shall apply. In addition, the logical unit shall clear the associated CA condition upon sending sense data by means of the autosense mechanism described in 5.8.4.3.~~

~~While the SCSI-2 rules for clearing the CA condition are in effect, a logical unit that supports the CLEAR ACA task management function shall ignore all CLEAR ACA requests and shall return a service response of FUNCTION COMPLETE (see 6.4).~~

~~If the logical unit accepts a value of one for the NACA bit and this bit was set to one in the CONTROL byte of the faulting command, then the SCSI-2 rules for clearing a CA condition shall not apply. In this case, the~~An ACA condition shall only be cleared:

a) ~~A~~as the result of a power on or a logical unit reset (see 5.8.7);

b) ~~Through~~by a CLEAR ACA task management function ~~issued~~ from ~~by~~ the ~~faulting~~ faulted initiator ~~as described in 6.4~~;

c) ~~T~~through a ~~Preempt and Clear~~PREEMPT AND ABORT service action of a PERSISTENT RESERVE OUT command that clears the tasks of the faulting initiator (see SPC-~~2~~3). If the PERSISTENT RESERVE OUT command is from the faulted initiator, it has the ACA attribute set; if it is from another initiator, it does not have the ACA attribute set; or
d) when a~~A~~ command with the ACA attribute terminates with a CHECK CONDITION status.

The state of all tasks in the task set when an ACA condition is cleared shall be modified as described in clause 7.

5.8.3 Incorrect Logical Unit selection

…
d) The target supports the logical unit but is incapable of determining if the peripheral device is attached or is not operational when it is not ready.

In response to an INQUIRY command the target shall return the INQUIRY data with the peripheral qualifier set to the value specified in SPC-3~~2~~. In response to a REQUEST SENSE command the target shall return the REQUEST SENSE data with a sense key of NO SENSE unless an ACA exists.

The target's response to any other command is vendor specific.

5.8.4.3 Autosense [as modified by 00-359]

Autosense is the automatic return of sense data to the application client coincident with the completion of a SCSI command ~~under the conditions described below. Inclusion of autosense support in a SCSI protocol standard is optional.~~Except for the SCSI Parallel Interface with information unit transfers disabled (see SPI-4), all protocols shall support autosense.

If supported by the protocol and logical unit and requested by the Execute Command remote procedure call (see 5.1), the device server shall only return sense data in this manner coincident with the completion of a command with a status of CHECK CONDITION. After autosense data is sent, sense data except sense data associated with a unit attention condition when the UAINTLCK bit equals one and the CA ~~(NACA equals zero)~~, if any, shall then be cleared. Autosense shall not affect ACA ~~(NACA equals one), see 5.8.1,~~ or sense data associated with a unit attention condition when the UAINTLCK bit equals one.

5.8.5 Unit Attention condition [as modified by 00-359]

Each logical unit shall generate a unit attention condition whenever the logical unit has been reset as described in 5.8.7 or by a power-on reset. In addition, a logical unit shall generate a unit attention condition for each initiator whenever one of the following events occurs:
a) ~~A~~ a removable medium may have been changed;
b) ~~T~~the mode parameters in effect for this initiator have been changed by another initiator;
c) ~~T~~the version or level of microcode has been changed;
d) ~~T~~tasks for this initiator were cleared by another initiator;
e) INQUIRY data has been changed;
f) ~~T~~the logical unit inventory has been changed;
g) ~~T~~the mode parameters in effect for the initiator have been restored from nonvolatile memory;
h) ~~A~~a change in the condition of a synchronized spindle; or
i) ~~A~~any other event requiring the attention of the initiator.

Logical units may queue unit attention conditions. After the first unit attention condition is cleared, another unit attention condition may exist (e.g., a power on condition followed by a microcode change condition).

A unit attention condition shall persist on the logical unit for each initiator until that initiator clears the condition as described in the following paragraphs.

If an INQUIRY command enters the enabled task state, the logical unit shall perform the INQUIRY command and shall neither report nor clear any unit attention condition.

If a REPORT LUNS command enters the enabled task state, the logical unit shall perform the REPORT LUNS command and shall not report any unit attention condition. The logical unit shall clear any unit attention condition established in response to a change in the logical unit inventory for all logical units for the initiator that sent the REPORT LUNS command. The logical unit shall not clear any other unit attention condition.

If a REQUEST SENSE command enters the enabled task state while a unit attention condition exists for the initiator that sent the REQUEST SENSE command, then the logical unit shall either:
a) Report any pending sense data and preserve all unit attention conditions on the logical unit; or,
b) Report a unit attention condition for the initiator that sent the REQUEST SENSE command. The logical unit may discard any pending sense data and shall clear the reported unit attention condition for that initiator.

If the logical unit has already generated the ACA or CA condition for a unit attention condition, the logical unit shall report the unit attention condition (the second action above).

If a command other than INQUIRY, REPORT LUNS, or REQUEST SENSE enters the enabled task state while a unit attention condition exists for the initiator that sent the command, the logical unit shall not perform the command and shall report CHECK CONDITION status. The logical unit shall provide sense data that reports a unit attention condition for the initiator that sent the command.

If a logical unit reports a unit attention condition with autosense or with an asynchronous event report, and the unit attention interlock (UAINTLCK) bit in the logical unit's control mode page is zero, then the logical unit may clear the reported unit attention condition for that initiator on the logical unit (see 5.8.4.2, 5.8.4.3 and SPC-3). If the unit attention interlock (UAINTLCK) bit is one, the logical unit shall not clear unit attention conditions reported with autosense or an asynchronous event report.

5.8.6 Hard reset

…

To process a logical unit reset the logical unit shall:

a) ~~A~~abort all tasks as described in 5.6; and

b) ~~C~~clear an ACA ~~(NACA equals one, see 5.2.3)~~ or CA ~~(NACA equals zero)~~ condition, if one is present;

…

6 Task Management Functions

6.1 Introduction

…

Table 24 — Task Management Functions

CLEAR ACA	I_T_L	6.4

…

The task manager response to task management requests is subject to the presence of access restrictions, as managed by ACCESS CONTROL IN and ACCESS CONTROL OUT commands (see SPC-3), as follows:

a) a task management request of ABORT TASK, ABORT TASK SET or CLEAR ACA shall not be affected by the presence of access restrictions;

6.2 ABORT TASK

…

The task manager shall abort the specified task if it exists. Previously established conditions, including MODE SELECT parameters, reservations, ACA, and CA shall not be changed by the ABORT TASK function.

6.3 ABORT TASK SET

…

The task manager shall perform an action equivalent to receiving a series of ABORT TASK requests. All tasks from that initiator in the task set serviced by the logical unit shall be aborted. Tasks from other initiators or in other task sets shall not be aborted. ~~A~~ CA ~~(NACA equals zero)~~ shall be cleared by the ABORT TASK SET function from the faulted initiator**.**
Other p~~P~~reviously established conditions, including MODE SELECT parameters, reservations, and ACA shall not be changed by the ABORT TASK SET function.

6.4 CLEAR ACA

Function Call
Service response = CLEAR ACA (IN (I_T_L Nexus))
Description:
This function shall be supported by a logical unit if it ~~accepts a NACA bit value of one in the CDB CONTROL byte (see 5.2.3)~~supports ACA.

The initiator invokes CLEAR ACA to clear an ACA condition from the task set serviced by the logical unit according to the rules specified in 5.8.1.2. For tasks with the ACA attribute (see 7.6.4) receipt of an CLEAR ACA function shall have the same effect as receipt of an ABORT TASK function (see 6.2). If successful, this function shall be terminated with a service response of FUNCTION COMPLETE.

If the task manager clears the ACA condition, any task within that task set may be completed subject to the rules for task set management specified in clause 7.

6.5 CLEAR TASK SET

…

Previously established conditions, including MODE SELECT parameters, reservations, and ACA ~~(NACA equals one, see 5.2.3)~~ shall not be changed by the CLEAR TASK SET function. A CA ~~(NACA equals zero)~~ shall be cleared by the CLEAR TASK SET function.

7 Task Set Management

7.1 Introduction to task set management

…

The rules for task set management only apply to a task after it has been entered into a task set. A task shall be entered into a task set unless a condition exists that causes that task to be completed with a status of BUSY, RESERVATION CONFLICT, TASK SET FULL, ACA ACTIVE or CHECK CONDITION when caused by the detection of an overlapped command. A task may also be completed because of a CHECK CONDITION status caused by certain protocol specific errors.

7.3 Controlling task set management

…The basic task management model requires the following task set management behaviors:
a) The only task attribute supported shall be SIMPLE;
b) The device server may reorder the actual processing sequence of tasks in any manner. Any data integrity exposures related to task sequence order shall be explicitly handled by the application client using the appropriate commands;
c) All the tasks in the task set that are in the blocked task state shall be aborted after an ACA or CA condition is cleared;
d) It shall not be possible to disable tagged queuing; and
e) Support for the CLEAR TASK SET task management function is optional.

7.4 Task management events

The following describe the events that drive changes in task state.

All older tasks ended:

All tasks have ended that were accepted into the task set earlier in time than the referenced task.

All older Head of Queue and older Ordered tasks ended:
All Head of Queue and Ordered tasks have ended that were accepted into the task set earlier in time than the referenced task.

ACA: An ACA condition has occurred ~~(NACA equals one, see 5.2.3)~~.

CA: An CA condition has occurred ~~(NACA equals zero)~~.

task abort: A task has been aborted as described in 5.6.

task completion: The device server has sent a service response of TASK COMPLETE for the task (see clause 5 and 5.5).

task ended: A task has completed or aborted.

ACA cleared: An ACA condition has been cleared.

CA cleared: An CA condition has been cleared.

7.5 Task states

…

7.5.3 Blocked task state

A task in the blocked task state is prevented from completing due to an ACA or CA condition. A task in this state shall not become a current task. While a task is in the blocked task state, any information the logical unit has or accepts for the task shall be suspended. If the TST field in the Control mode page (see SPC-2) equals 000b the blocked task state is independent of the initiator. If the TST field equals 001b the blocked task state applies only to the faulted initiator.

7.6 Task Attributes

…

7.6.4 ACA Task

A task having the ACA attribute shall be accepted into the task set in the enabled task state. There shall be no more than one ACA task per task set (see 5.8.1.1).

7.7 Task state transitions

[CA and ACA appear in figure 33]

Transition S0:S1 (Ordered Task): Provided an ACA or a CA condition does not exist or if the TST field contains 001b in the Control mode page (see SPC-2) provided the task is not for the faulted initiator and the QERR field is not 01b in the Control mode page, a dormant task having the ORDERED attribute shall enter the enabled task state when all older tasks have ended. If the TST field contains 000b in the Control mode page, this transition shall not occur while an ACA or a CA condition is in effect for any initiator. If the TST field contains 001b in the Control mode page, this transition shall not occur while an ACA or a CA condition is in effect for the initiator that created the ordered task.

Transition SO:S1 (Simple task): Provided an ACA or a CA condition does not exist or if the TST field contains 001b in the Control mode page (see SPC-2) provided the task is not for the faulted initiator, a dormant task having the SIMPLE attribute shall enter the enabled task state when all older Head of Queue and older Ordered tasks have ended. If the TST field contains 000b in the Control mode page, this transition shall not occur while an ACA or a CA condition is in effect for any initiator. If the TST field contains 001b in the Control mode page, this transition shall not occur while an ACA or a CA condition is in effect for the initiator that created the simple task.

Transitions S0:S3, S2:S3: A task abort event shall cause the task to unconditionally enter the ended task state.

Transition S1:S2: If the TST field contains 000b in the Control mode page, an ACA or a CA condition shall cause an enabled task to enter the blocked task state. If the TST field contains 001b in the Control mode page, an ACA or a CA condition shall cause an enabled task for the faulted initiator to enter the blocked task state.

Transition S1:S3: A task that has completed or aborted shall enter the ended task state. This is the only state transition that applies to an ACA task.

Transition S2:S1: When an ACA or a CA condition is cleared and the QERR field is 00b in the Control mode page, a blocked task shall re-enter the enabled task state. When an ACA or a CA condition is cleared and the QERR field is 11b in the Control mode page, a blocked task for other than the faulting initiator shall re-enter the enabled task state.

7.8 Task set management examples

7.8.1 Introduction

…

A task set is shown as an ordered list or queue of tasks with the head of the queue towards the top of the page. A new Head of Queue task always enters the task set at the head, displacing older Head of Queue tasks. Simple, Ordered and ACA tasks always enter the task set at the end of the queue.

…

Task attributes:

a) Simple tasks -- rounded corners;
b) Ordered -- square corners and thin lines;
c) Head of Queue -- square corners and thick lines; or
d) ACA tasks -- square corners and thin dashed lines.

The conditions preventing a dormant task from entering enabled task state (except for ACA and CA conditions) are shown by means of “blocking boundaries”. Such boundaries appear as horizontal lines with an arrow on both ends. The tasks causing the barrier condition are described as part of each example. A task is impeded by the barrier if it is between the boundary and the end of the queue. When no ACA or CA is in effect, a task enters the enabled task state after all intervening barriers have been removed.

7.8.4 ACA task

Figure 37 shows the effects of an ACA condition on the task set. This example assumes the QERR field is set to 00b in the Control mode page (see SPC-~~2~~3). Consequently, clearing an ACA condition does not cause tasks to be aborted.

[figure 37 - ACA task example]

The completion of task 2 with CHECK CONDITION status causes task 1 to enter the blocked task state shown in snapshot 2. In snapshot 3, Ordered task 3 is aborted using the ABORT TASK task management function and ACA task 5 is created to perform additional handling for the exception. Once the ACA condition is cleared, (snapshot 4) Simple task 1 is allowed to reenter the enabled task state. Since there are no Head of Queue or Ordered tasks older than task 4, it too is allowed enter the enabled task state.

Suggested changes to SPC-3

SCSI Protocols:

…

~~Serial Storage Architecture SCSI-2 Protocol SSA-S2P [ANSI X3.294:1996]~~
Serial Storage Architecture SCSI-3 Protocol SSA-S3P [ANSI NCITS.309:1998]

~~Common Access Method:~~
~~SCSI Common Access Method CAM [ISO/IEC 9316-421] [ANSI X3.232:1996]~~

The term SCSI is used to refer to the family of standards described in this clause. ~~The Small Computer System Interface - 2 standard (ANSI X3.131-1994) and the architecture that it describes are referred to herein as SCSI-2.~~

2.2 Approved references

…

ISO/IEC 60027-2-am2 (1999-01), Letter symbols to be used in electrical technology - Part 2: Telecommunications and electronics (Amendment 2)

~~*ISO/IEC 9316:1995-11,* Small Computer System Interface -2 standard, (SCSI-2) [ANSI X3.270:1996]~~

3.1.6 auto contingent allegiance (ACA): One of the conditions of a task set following the return of a CHECK CONDITION status. A detailed definition of ACA may be found in SAM-2.

3.1.13 contingent allegiance (CA): One of the conditions of a task set following the return of a CHECK CONDITION status. A detailed definition of CA may be found in ~~SCSI-2~~SAM-2.

3.1.76 task set: A group of tasks within a logical unit, whose interaction is dependent on the task management (queuing), CA, and ACA rules. See SAM-2 and the Control mode page (see 8.3.6).

3.2 Acronyms

ACA Auto Contingent Allegiance (see 3.1.6)
CA Contingent Allegiance (see 3.1.13~~ and SCSI-2~~)
SCSI The architecture defined by the family of standards described in clause 1
~~SCSI-2 The architecture defined by the Small Computer System Interface - 2 standard (see 2.2)~~

5.5.3.6.4 Preempting an existing persistent reservation with the PREEMPT AND ABORT service action

…

c) 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 the TST field is 000b (see 8.3.6) and ACA or CA conditions exist for initiators other than the initiator being preempted, the PERSISTENT RESERVE OUT command shall be terminated prior to processing with a status of ACA ACTIVE if the NACA bit equals one ~~(see SAM-2)~~ or BUSY if the NACA equals zero. If the TST field contains 001b, then ACA or CA conditions for initiators other than the initiator being preempted shall not prevent the ~~execution~~ processing of the PERSISTENT RESERVE OUT command; and

7.3.1 INQUIRY command introduction

…

NOTE 11 - An application client may receive a CHECK CONDITION status response with the sense key set to ILLEGAL REQUEST upon sending an INQUIRY command with the CMDDT bit set to one to some ~~SCSI-2~~old device servers, since this bit was reserved in ~~SCSI-2~~previous versions of the SCSI standards.

…

The Normal ACA Supported bit (NORMACA) of one indicates that the device server supports setting the NACA bit to one in the CONTROL byte of the CDB (see SAM-2). A NORMACA bit of zero indicates that the device server does not support setting the NACA bit to one.

Table 48 — Peripheral device type (part 1 of 2)
06h ~~SCSI-2~~ Scanner device (obsolete)
09h ~~SCSI-2~~ Communications device (obsolete)

Table 49 — Version

Code	Description
00h	The device does not claim conformance to any standard.
02h	~~The device complies to ANSI X3.131:1994.~~Obsolete
03h	The device complies to ANSI X3.301:1997 (SPC).
04h	The device complies to ~~this standard.~~ANSI xyz:2001 (SPC-2)
05h	The device complies to this standard (SPC-3).
80h	~~The device complies to ISO/IEC 9316:1995.~~obsolete
82h	~~The device complies to ISO/IEC 9316:1995 and to ANSI X3.131:1994.~~obsolete
83h	~~The device complies to ISO/IEC 9316:1995 and to ANSI X3.301:1997.~~obsolete
84h	~~The device complies to ISO/IEC 9316:1995 and to this standard.~~obsolete

~~ANSI X3.131:1994 is SCSI-2~~
~~ANSI X3.301:1997 is SPC~~
~~ISO/IEC 9316:1995 is SCSI-2~~

7.20 REQUEST SENSE command
7.20.1 REQUEST SENSE command introduction
…
If the device server is in the standby power condition or idle power condition when a REQUEST SENSE command is received and there is no ACA or CA condition, the device server shall return a sense key of NO SENSE and an additional sense code of LOW POWER CONDITION ON. On completion of the command the logical unit shall return to the same power condition that was active before the REQUEST SENSE command was received. A REQUEST SENSE command shall not reset any active power condition timers.

8.3.6 Control mode page
Table 162 — Queue error management (QERR) field
Value Definition
00b Blocked tasks in the task set shall resume after an ACA or CA condition is cleared (see SAM-2).

Table 109 — ASC and ASCQ assignments
Table C.1 — ASC and ASCQ assignments
Table C.2 — Operation Codes
Table C.3 — Log Page Codes
Table C.4 — Mode Page Codes
S - SCANNER DEVICE (~~SCSI-2~~obsolete)
C - COMMUNICATION DEVICE (~~SCSI-2~~obsolete)

[omitted in revision 0 of this proposal:]
10 Commands for processor type devices
10.3 SEND command
~~If the SCSI-3 bit is zero, then the AEN data format, as defined by the SCSI-2 standard, shall be used. If the SCSI-3 bit is one, then the AER data format shown in table 197 shall be used.~~
The SCSI-3 bit shall be set to one. A SCSI-3 bit value of zero is obsolete.

A Procedures for logging operations in SCSI
A.5 Exception condition during logging
…

The pseudocode in A.5.1 through A.5.3 assumes that ACA is implemented and requested in the CDB control byte. ~~If this is not the case, the implementation may be based on the SCSI-2 TIB[1] or other applicable standards.~~

…

~~1. TIB for IT - Procedures for Logging Operations (X3-131-1994/TIB-1).~~

Wording from SCSI-2 for reference
[excluding "extended contingent allegiance" references]

3.1.8 contingent allegiance: A condition typically generated by a CHECK CONDITION status during which a target preserves sense data (see 7.6.).

7.3.2 CHECK CONDITION: This status indicates that a contingent allegiance condition has occurred (see 7.6).
7.3.8 COMMAND TERMINATED: This status shall be returned whenever the target terminates the current I/O process after receiving a TERMINATE I/O PROCESS message (see 6.6.22). This status also indicates that a contingent allegiance condition has occurred (see 7.6).

7.5.2 Incorrect initiator connection
…
An incorrect initiator connection also occurs on an initial connection when an initiator:
a) attempts to establish an I_T_L_Q nexus when an I_T_L nexus already exists from a previous connection, or
b) attempts to establish an I_T_L nexus when an I_T_L_Q nexus already exists, unless there is a contingent allegiance or extended contingent allegiance condition present for the logical unit.

7.5.3 Selection of an invalid logical unit
The target's response to selection of a logical unit that is not valid is described in the following paragraphs.

The logical unit may not be valid because:
…
d) the target supports the logical unit but is incapable of determining if the peripheral device is attached or is not operational when it is not ready. In response to an INQUIRY command, the target shall return the INQUIRY data with the peripheral qualifier set to the value required in 8.2.5.1. In response to a REQUEST SENSE command, the target shall return the REQUEST SENSE data with a sense key of NO SENSE unless a contingent allegiance exists. The target's response to any other command is vendor-specific.

7.6 Contingent allegiance condition
The contingent allegiance condition shall exist following the return of CHECK CONDITION or COMMAND TERMINATED status. The contingent allegiance condition shall be preserved for the I_T_x nexus until it is cleared.

The contingent allegiance condition shall be cleared upon the generation of a hard reset condition, or by an ABORT message, a BUS DEVICE RESET message, or any subsequent command for the I_T_x nexus. While the contingent allegiance condition exists the target shall preserve the sense data for the initiator.

While the contingent allegiance condition exists, if the target is unable to maintain separate sense data, the target shall respond to any other requests for access to the logical unit or target routine from another initiator with a BUSY status.

Execution of all tagged I/O processes for the I_T_L nexus for which the contingent allegiance condition exists shall be suspended until the contingent allegiance condition is cleared.

7.8 Queued I/O processes
The implementation of queuing for I/O processes is optional. Queuing of I/O processes allows a target to accept multiple I/O processes.

There are two methods for implementation of queuing, tagged and untagged. Tagged queuing allows a target to accept multiple I/O processes from each initiator for each logical unit. Untagged queuing allows a target to accept one I/O process from each initiator for each logical unit or target routine. Untagged queuing may be supported by SCSI-1 or SCSI-2 devices. Tagged queuing is new in SCSI-2.

A target may support both tagged and untagged queuing. An initiator may not mix the use of tagged and untagged queuing for I/O processes to a logical unit, except during a contingent allegiance or extended contingent allegiance condition when only untagged initial connections are allowed. An initiator that elects to use tagged queuing does not preclude another initiator on the same SCSI bus from using untagged queuing.

7.8.1 Untagged queuing

…

An I/O process received without a queue tag message, while there are any tagged I/O commands in the command queue from the same initiator, is an incorrect initiator connection (see 7.5.2), unless there is a contingent allegiance or extended contingent allegiance condition.

…

There are two methods of dealing with the command queue following a contingent allegiance condition. The method used is specified in the control mode page by the queue error management bit (see 8.3.3.1).

The first method allows the execution of tagged I/O processes to resume when the contingent allegiance or extended contingent allegiance condition is cleared. The target returns BUSY status to other initiators while the contingent allegiance or extended contingent allegiance condition exists. During this time, all tagged I/O processes are suspended. All I/O processes used for recovery operations shall be untagged. The queue may be modified by removing all or selected I/O processes from the queue as part of the recovery procedure.

If commands are combined by the queuing algorithm such that the exception condition affects more than one command, then a contingent allegiance condition shall be generated for all affected commands.

The second method aborts all I/O processes when the contingent allegiance or extended contingent allegiance condition is cleared. A unit attention condition shall be generated for all other initiators and the additional sense code shall be set to COMMANDS CLEARED BY ANOTHER INITIATOR.

A device that does not support tagged I/O processes (e.g. not implemented, disabled by the Dque bit in the control mode page, or switched to an operating definition that does not include tagged I/O processes) it shall respond to any queue tag message with a MESSAGE REJECT message. The target shall continue the I/O process as if it were an untagged I/O process.

Tagged queuing may also be temporarily disabled during certain initialization periods or to control internal resource utilization. During these periods the target may return QUEUE FULL status or it may respond to any queue tag message with a MESSAGE REJECT message.

Several messages may be used to clear part or all of the command queue. Please refer to the ABORT, ABORT TAG, BUS DEVICE RESET, and CLEAR QUEUE messages in clause 6 for details.

7.9 Unit attention condition

…

If an INQUIRY command is received from an initiator to a logical unit with a pending unit attention condition (before the target generates the contingent allegiance condition), the target shall perform the INQUIRY command and shall not clear the unit attention condition. If the INQUIRY command is received after the target has generated the contingent allegiance condition for a pending unit attention condition, then the unit attention condition on the logical unit shall be cleared, and the target shall perform the INQUIRY command.

If any other command is received after the target has generated the contingent allegiance condition for a pending unit attention condition, the unit attention condition on the logical unit shall be cleared, and if no other unit attention condition is pending the target shall perform the command. If another unit attention condition is pending, the target shall not perform the command and shall generate another contingent allegiance condition.

If a REQUEST SENSE command is received from an initiator with a pending unit attention condition (before the target generates the contingent allegiance condition), then the target shall either:

a) report any pending sense data and preserve the unit attention condition on the logical unit, or,
b) report the unit attention condition, may discard any pending sense data, and clear the unit attention condition on the logical unit for that initiator.

If the target has already generated the contingent allegiance condition for the unit attention condition, the target shall perform the second action listed above.

If an initiator issues a command other than INQUIRY or REQUEST SENSE while a unit attention condition exists for that initiator (prior to generating the contingent allegiance condition for the unit attention condition), the target shall not perform the command and shall report CHECK CONDITION status unless a higher priority status as defined by the target is also pending (e.g. BUSY or RESERVATION CONFLICT).

If, after generating the contingent allegiance condition for a pending unit attention condition, the next command received from that initiator on the logical unit is not REQUEST SENSE, then that command shall be performed and the unit attention condition shall be cleared for that initiator on the logical unit and the sense data is lost (see 7.6).

8.1.2.2 Using the REQUEST SENSE command

Whenever a contingent allegiance condition (see 7.6) is established, the initiator that received the error should issue a REQUEST SENSE command to receive the sense data describing what caused the contingent allegiance condition.

If the initiator issues some other command, the sense data is lost.

8.2.14 REQUEST SENSE Command

…

The sense data:
a) shall be available if a contingent allegiance condition exists for the I_T_x nexus;
b) shall be available if other information (e.g. medium position) is available in any field;
c) may be available if an unexpected disconnect occurred.

8.3.3.1 Control mode page

…

A queue error management (QErr) bit of zero specifies that remaining suspended I/O process shall resume after the contingent allegiance condition or extended contingent allegiance condition (see 7.8).

A QErr bit of one specifies all remaining suspended I/O processes shall be aborted after the contingent allegiance condition or extended contingent allegiance condition (see 7.8). A unit attention condition (see 7.9) shall be generated for each initiator that had a suspended I/O process aborted except for the initiator that had the contingent allegiance condition or extended contingent allegiance condition. The target shall set the additional sense code to TAGGED COMMANDS CLEARED BY ANOTHER INITIATOR.