⑤ Seagate

Seagate Technology   Tel: 405-324-3070
10323 West Reno (West Dock)   Fax: 405-324-3794
Oklahoma City, OK 73127-9705   gene_milligan@notes.seagate.com
P.O. Box 12313
Oklahoma City, OK 73157-2313

Date: 9/21/97

To: X3T10 Membership

From: Gene Milligan, T10 Principal member

Subject: **Proposal for Contingent Allegiance / Auto Contingent Allegiance Handling**

Proposal Status:

Revision 0 included an overview of the key functional attributes of the proposal. Revision 1 incorporates the conclusions of the 9/97 T10 working group. In particular the SEXE bit was replaced with a Task Set Type (TST) field which allows that error handling behavior to be tied to the type of task set. Thus the former SEXE=One behavior is replaced by the handling of TST=001b(task set per initiator per LUN). With this type ordering and task set blocking is on a per initiator basis. TST=000b(task set per LUN for all initiators) is equivalent to the SEXE=Zero behavior of Revision 0 which in turn is equivalent to SPC and SAM prior to the dash two versions.

CA_ACA Proposal:

The following proposal uses snippets of the clauses to keep sufficient context. Changes are denoted by: addition; <<deletion>>; and **((proposer s comment which is not part of the proposal nor part of the draft standard)).**

Changes to SPC-2:

3.1.5 auto contingent allegiance: <<The>> One of the conditions of a task set following the return of a CHECK CONDITION or COMMAND TERMINATED status. A detailed definition of ACA may be found in SAM.

3.1.TBD contingent allegiance: One of the conditions of a task set following the return of a CHECK CONDITION or COMMAND TERMINATED status. A detailed definition of CA may be found in SCSI-2.

3.1.TBD task set: A group of tasks within a logical unit, whose interaction is dependent on the task management (queuing), CA, and ACA rules. <<defined in SAM >> See SAM and the Control Mode Page (8.3.4).

CA   Contingent Allegiance (see 3.1.TBD and SCSI-2)

7.13.1.6 Preempt and Clear

The Preempt and Clear service action shall clear any CA or ACA condition associated with the initiator being preempted and shall clear any tasks with an ACA attribute **((the plural in this requirement was in violation of SAM prior to this proposal))** from that initiator. If TST=000b CA or ACA conditions for other initiators shall prevent the execution of the PERSISTENT RESERVE OUT task, which shall end with a status of BUSY (NACA = 0) or ACA ACTIVE (NACA = 1).

> NOTE 28 The Preempt and Clear service action will clear the ACA condition associated with the initiator being preempted even though the task is terminated with an ACA ACTIVE status. Thus, the next command arriving at the device server will not encounter the ACA condition previously active for the initiator being preempted.

7.20 REQUEST SENSE command

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 CA or ACA 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.

7.20.3 Deferred errors

"If the task terminates with CHECK CONDITION status and the subsequent sense data returns a deferred error that task shall not have been executed. After the device server detects a deferred error condition, it shall return a deferred error according to the rules described below:

a) If no external system intervention is necessary to recover a deferred error, a deferred error indication shall not be posted unless required by the error handling parameters of a MODE SELECT command. The occurrence of the error may be logged if statistical or error logging is supported.

b) If it is possible to associate a deferred error with a causing initiator, a task set, and with a particular function or a particular subset of data, and the error is either unrecovered or required to be reported by the mode parameters, a deferred error indication shall be returned to an application client on the causing initiator. If an application client on an initiator other than the causing initiator attempts access to the particular function or subset of data associated with the deferred error and the TST=000b, <<a BUSY status shall be returned to that application client in response to>> the command attempting the access shall be responded to according to the requirements in SAM.

c) If a deferred error cannot be associated with a causing initiator or with a particular subset of data, the device server shall return a deferred error indication to an application client on each initiator. If multiple deferred errors have accumulated for some initiators, only the last error shall be returned.

d) If a deferred error cannot be associated with a particular logical unit, the device server shall return a deferred error indication to an application client associated with any logical unit on the appropriate initiator.

If a task has never entered the enabled task state, and a deferred error occurs, the task shall be terminated with CHECK CONDITION status and deferred error information posted in the sense data. If a deferred error occurs after a task has entered the enabled task state and the task is

affected by the error, the task shall be terminated by CHECK CONDITION status and the current error information shall be returned in the sense data. In this case, if the current error information does not adequately define the deferred error condition, a deferred error may be returned after the current error information has been recovered. If a deferred error occurs after a task has entered the enabled task state and the task completes successfully, the device server may choose to return the deferred error information after the completion of the current command in conjunction with a subsequent command that has not started execution."

8.3.4 Control mode page

The control mode page (see table 98) provides controls over several SCSI-3 features that are applicable to all device types such as tagged queuing, asynchronous event reporting, and error logging.

Table 98 _ Control mode page

Bit Byte	7	6	5	4	3	2	1	0
0	PS	Reserved		Page code (0Ah)				
1				Page length (0Ah)				
2		TST		Reserved			GLTSD	RLEC
3	Queue algorithm modifier			Reserved		QErr		DQue
4	Reserved	RAC	Reserved		SWP	RAERP	UAAERP	EAERP
5				Reserved				
6	(MSB)							
				Ready AER holdoff period				
7								(LSB)
8	(MSB)							
				Busy timeout period				
9								(LSB)
10				Reserved				
11				Reserved				

A task set type field (TST) specifies the type of task set supported by the LUN and if supported selected by the Initiator. The TST field, if changeable, shall reflect in all initiator pages the state selected by the most recent MODE SELECT by any initiator. If the most recent MODE SELECT by any initiator changes the setting of this field the device server shall establish a unit attention condition for all initiators except the one that issued the MODE SELECT command (see SAM). The device server shall set the additional sense code to MODE PARAMETERS CHANGED. The TST settings are:

Value	Definition
000b	Task set per LUN for all initiators
001b	Task set per Initiator per LUN
Other values	Reserved

The queue error management (QErr) field specifies how the device server shall handle <<blocked>> tasks when another task receives a COMMAND TERMINATED or CHECK CONDITION status (see table n1).

Table n1 - Queue error management (QErr)

Value	Definition
00b	Blocked tasks in the task set shall resume after CA or an ACA condition is cleared (see SAM).
01b	All the blocked tasks in the task set shall be aborted when the COMMAND TERMINATED or CHECK CONDITION status is sent. A unit attention condition (see SAM) shall be generated for each initiator that had blocked tasks (TST=000b) aborted except for the initiator to which the COMMAND TERMINATED or CHECK CONDITION status was sent. The device server shall set the additional sense code to COMMANDS CLEARED BY ANOTHER INITIATOR.
10b	Reserved
11b	Blocked tasks in the task set belonging to the initiator to which a COMMAND TERMINATED or CHECK CONDITION status is sent shall be aborted when the status is sent.

A disable queuing (DQue) bit of zero specifies that tagged queuing shall be enabled if the device server supports tagged queuing. A DQue bit of one specifies that tagged queuing shall be disabled. Any queued commands received by the device server shall be aborted. The method used to abort queued commands is protocol-specific.

Changes to SAM-2:

3.1.94 task set: A group of tasks within a <<target device>> LUN, whose interaction is dependent on the queuing and contingent allegiance (NACA = 0) or auto contingent allegiance (NACA = 1) rules of clause 7.

4.7.4 Logical Unit

(Replace Task Set with Task Set(s) several places.)

Object definition 6: Logical Unit
Logical Unit = Device server + Logical Unit Number + (Logical Unit) + Task Set(s)

Task Set : A set of tasks whose interaction is determined by the rules for task set management specified in clause 7 and the auto contingent allegiance rules specified in subclause 5.6.1. As defined in object definition 6, there shall be one task set per logical unit for all initiators or one task set per initiator.

5.4 Task and Command Lifetimes
This clause specifies the events delimiting the beginning and end of a task or pending SCSI-3 command from the viewpoint of the device server and application client.
The device server shall create a task upon receiving an SCSI Command Received indication unless the command represents a continuation of a linked command as described in clause 5.
The task shall exist until:
a) The device server sends a protocol service response for the task of TASK COMPLETE.
b) A power on condition occurs.
c) The logical unit executes a logical unit reset operation as described in 5.6.7.
d) The task manager executes an ABORT TASK referencing the specified task

e) The task manager executes an ABORT TASK SET, CLEAR ALL TASK SETS or a CLEAR TASK SET task management function directed to the task set containing the specified task.

----

e) A service response of FUNCTION COMPLETE following an ABORT TASK SET, CLEAR ALL TASK SETS or a CLEAR TASK SET task management function directed to the task set containing the specified task.

5.6.1.1 Logical Unit Response to Auto Contingent Allegiance

The contingent allegiance (NACA = 0) or auto contingent allegiance (NACA = 1) condition shall not cross task set boundaries and shall be preserved until it is cleared as described in 5.6.1.2. If requested by the application client and supported by the protocol and logical unit, sense data shall be returned as described in 5.6.4.2.

Notes:
1. The SCSI-2 contingent allegiance condition has had an alternate added and the extended auto contingent allegiance condition <<have>> has been replaced in SCSI-3 by auto contingent allegiance in conjunction with the NACA bit.
2. If the SCSI-3 protocol does not enforce state synchronization as described in 4.6.1, there may be a time delay between the occurrence of the contingent or auto contingent allegiance condition and the point at which the initiator becomes aware of the condition.

After sending status and 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 auto contingent allegiance condition is in effect may be entered into the faulted task set under the conditions described below. **((Order rearranged))**

> As described in 5.6.1.2, the setting of the NACA bit in the control byte of the faulting command determines the rules that apply to an ACA condition caused by that command. If the NACA bit was set to zero the SCSI-2 contingent allegiance rules shall apply. In that case, the completion of a subsequent command from the faulted initiator with a status of CHECK CONDITION or COMMAND TERMINATED shall clear the existing contingent allegiance and cause a new contingent allegiance or auto contingent allegiance condition to exist. The rules for responding to the new <<auto contingent allegiance>> condition shall be determined by the state of the NACA bit in the new faulted command. If the NACA bit was set to one in the CDB control byte of the faulting command, then a new task created while the ACA condition is in effect shall not be entered into the faulted task set <<provided>> unless all of the following conditions are true:
> a) The command was originated by the faulted initiator,
> b) The task has the ACA attribute,
> c) No other task having the ACA attribute is in the task set.
>
> If 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 auto contingent allegiance condition is in effect for the task set or if the NACA bit was set to zero in the CDB for the faulting command, 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. As noted in 5.6.1.2, any existing contingent or auto contingent allegiance condition for that initiator shall be cleared and a new contingent or auto contingent allegiance condition shall be established.
>
> If TST=000b (See SPC-2): Except for a PERSISTENT RESERVE command with a Preempt and Clear action as described in subclause 5.6.1.2, tasks created by other

initiators while the ACA condition is in effect shall not be entered into the task set and shall be completed with a status of BUSY if NACA = 0 or ACA ACTIVE if NACA = 1. <<Tasks created by other initiators while the ACA condition is in effect shall not be entered into the faulted task set and shall be completed with a status of ACA ACTIVE.>>

If the TST=001b: As described in 5.6.1.2, the setting of the NACA bit in the control byte of the faulting command determines the rules that apply to an ACA condition caused by that command. If the NACA bit was set to zero the SCSI-2 contingent allegiance rules shall apply. In that case, the completion of a subsequent command from the faulted initiator with a status of CHECK CONDITION or COMMAND TERMINATED shall clear the existing contingent allegiance and cause a new contingent or auto contingent allegiance condition to exist. The rules for responding to the new <<auto contingent allegiance>> condition shall be determined by the state of the NACA bit in the new faulted command. If the NACA bit was set to one in the CDB control byte of the faulting command, then a new task created by the faulting initiator while the CA or ACA condition is in effect shall not be entered into the faulted task set <<provided>> unless all of the following conditions are true:
<<a) The command was originated by the faulted initiator,>>
b) The task has the ACA attribute,
c) No other task from the faulting initiator having the ACA attribute is in the task set.

If the task is from the faulting 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 from the faulting initiator having the ACA attribute is received and no contingent or auto contingent allegiance condition is in effect for the <<task set>> faulting initiator or if the NACA bit was set to zero in the CDB for the faulting command, 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. As noted in 5.6.1.2, any existing contingent or auto contingent allegiance condition for that initiator shall be cleared and a new contingent or auto contingent allegiance condition shall be established.

<<Except for a PERSISTENT RESERVE command with a Preempt and Clear action as described in subclause 5.6.1.2,>> Tasks in a task set for other initiators shall be enabled and completed in accordance with the requirements of Clause 7. Tasks created by other initiators while the ACA condition is in effect shall <<not>> be entered into<< the>> that initiator's task set provided that the task set is not full and the command is a supported command without an illegal parameter.<<and shall be completed with a status of ACA ACTIVE>>.

5.6.4.2 Autosense
Autosense is the automatic return of sense data to the application client coincident with the completion of an SCSI-3 command under the conditions described below. The return of sense data in this way is equivalent to an explicit command from the application client requesting sense data immediately after being notified that an ACA condition has occurred. Inclusion of autosense support in an SCSI-3 protocol standard is optional.

As specified in clause 5, the application client may request autosense service for any SCSI command. If supported by the protocol and logical unit and requested by the application client, the device server shall only return sense data in this manner coincident with the completion of a

command with a status of CHECK CONDITION or COMMAND TERMINATED. The sense data and the CA (NACA = 0) or ACA (NACA = 1) shall then be cleared.

5.6.5 Unit Attention 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 is received from an initiator to a logical unit with a pending unit attention condition (before the logical unit generates the contingent or auto contingent allegiance condition), the logical unit shall perform the INQUIRY command and shall not clear the unit attention condition.
If a request for sense data is received from an initiator with a pending unit attention condition (before the logical unit establishes the contingent or auto<<matic>> contingent allegiance condition), then the logical unit 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.

If the second option is chosen (reporting the unit attention condition), the logical unit may discard any pending sense data and may clear the unit attention condition for that initiator.

If the logical unit has already generated the contingent or auto contingent allegiance condition for the unit attention condition, the logical unit shall perform the second action listed above. If NACA for the REQUEST SENSE command is zero and the command is untagged the contingent allegiance condition shall be cleared.

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 or auto contingent allegiance condition for the unit attention condition), the logical unit shall not perform the command and shall report <<CHECK CONDITION>> BUSY (NACA = 0) or ACA ACTIVE (NACA = 1) status unless a higher priority status **((??))** as defined by the logical unit is also pending (see 50)**((SIC))**. **((Because of the ?? this change is still under construction since for NACA = 0 the specific command is not the trigger but whether it is tagged or untagged. Therefore this clause should remain an issue regardless of the outcome of the basic proposal. It appears to me this subclause has become confused as to whether it is describing a Unit Attention condition or an AEN without an error.))**

If a logical unit successfully sends an asynchronous event report informing the initiator of the unit attention condition, then the logical unit shall clear the unit attention condition for that initiator on the logical unit (see 5.6.4.1).

5.6.7 Logical Unit Reset

A logical unit reset is a response to a LOGICAL UNIT RESET task management request (see 6.5), or a some other logical unit reset event, such as a target hard reset (see 5.6.6). The definition of such events may be device-specific or <<dependant>> dependent on the protocol and interconnect. Each appropriate SCSI-3 standard shall specify the conditions under which a logical unit reset shall be executed.

To execute a logical unit reset the logical unit shall:

a) Abort all tasks in its task set(s);
b) Clear a<<n>> contingent allegiance (NACA = 0) or auto contingent allegiance (NACA = 1) condition, if one is present;

c) Release all SCSI device reservations;

d) Return the device's operating mode to the appropriate initial conditions, similar to those conditions that would be found following device power-on. The MODE SELECT conditions (see the SPC standard) shall be restored **((Although not part of this proposal the last sentence seems wrong. The words in SPC are "MODE SELECT parameters".))**

6 Task Management Functions

CLEAR ACA (Logical Unit Identifier) - Clear auto contingent allegiance condition. This function shall be supported if the logical unit accepts a<<n>> NACA bit value of one in the CDB control byte. <<and may be supported if the logical unit does not accept an NACA bit value of one in the CDB control byte>> (see 5.1.2).

All SCSI-3 protocol <<specifications>> standards shall provide the functionality needed for a task manager to implement all of the task management functions defined above.

The following task management functions are defined:

CLEAR ALL TASK SETS (Logical Unit Identifier) - Abort all tasks in all task sets for the specified <<task set>> logical unit. This function <<shall be supported by>> is optional. <<all logical units that support tagged tasks (see object definition 7) and may be supported by logical units that do not support tagged tasks>>.

CLEAR TASK SET (Logical Unit Identifier) - Abort all tasks in the specified task set. This function shall be supported by all logical units that support tagged tasks (see object definition 7) and may be supported by logical units that do not support tagged tasks.

LOGICAL UNIT RESET (Logical Unit Identifier) - Perform a logical unit reset as described in 5.6.7 by terminating all tasks in the task set(s) and propagating the reset to all nested logical units (See 4.7.5). Support for this function is mandatory for hierarchical logical units and may be supported by non-hierarchical logical units.

((Other places refer to 4.7.4 for nested logical units including 4.7.4 itself. While a definition of nested logical units does not seem to exist, 4.7.5 has a figure with that title.))

IMPLEMENTORS NOTES: The TARGET RESET, CLEAR ALL TASK SETS, CLEAR TASK SET, ABORT TASK and ABORT TASK SET functions provide a means to terminate one or more tasks prior to normal completion. <<The TARGET RESET command clears all tasks for all initiators on all task sets of the target. The CLEAR TASK SET function terminates all tasks for all initiators on the specified task set of the target. An ABORT TASK SET function terminates all tasks for the initiator on the specified task set of the target. An ABORT TASK function terminates only the specified task.>>

6.3 CLEAR ACA
Function Call
Service response = CLEAR ACA (Logical Unit Identifier)

Description:
This function shall only be implemented by a logical unit that accepts a<<n>> NACA bit value of one in the CDB control byte (see 5.1.2). **((Note the conflict prior to the above deletion in 6.))**

The initiator invokes CLEAR ACA to clear an auto contingent allegiance condition from the task set serviced by the logical unit according to the rules specified in 5.6.1.2. If successful this <<The>> function shall <<always>> be terminated with a service response of FUNCTION COMPLETE.

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

6.TBD CLEAR ALL TASK SETS
Function Call:
Service response = CLEAR ALL TASK SETS (Logical Unit Identifier)

Description:
This function may be supported by all logical units and initiators (see object definition 7).

If supported the LUN shall perform an action equivalent to receiving a series of ABORT TASK requests from each initiator.

All tasks, from all initiators, in the specified LUN shall be aborted. The medium may have been altered by partially executed commands. All pending status and data for that logical unit for all initiators shall be cleared.

No status shall be sent for any task. A unit attention condition shall be generated for all other initiators with tasks in the LUN. When reporting the unit attention condition the additional sense code shall be set to COMMANDS CLEARED BY ANOTHER INITIATOR.

Previously established conditions, including MODE SELECT parameters, reservations, and contingent or auto contingent allegiance shall not be changed by the CLEAR ALL TASK SETS function.

((Was it really intended that the CLEAR TASK SET be able to come through a contingent allegiance regardless of initiator or is this in conflict with other sections?))

((The CLEAR TASK SET message in SAM is defined to abort all tasks for all initiators in the specified task set. I believe the wording could have been construed to be the same as this function since no method was provided for identifying task sets other than by LUN number. However it seems more flexible to have two different messages to allow increased flexibility along with more expansive backward compatibility. But it may be adequate to define it such that ABORT TASK SET eliminates commands for a given initiator whether they be in one LUN s task set for all initiators or in one LUN s task set for just the current initiator with CLEAR TASK set clearing tasks for all initiators regardless of how they are partitioned within the LUN.))

6.4 CLEAR TASK SET
Function Call:
Service response = CLEAR TASK SET (Logical Unit Identifier)

Description:
This function shall be supported by all logical units that support tagged tasks (see object definition 7) and may be supported by logical units that do not support tagged tasks.

If the TST field equals 000b (See SPC-2) the target shall perform an action equivalent to receiving a series of ABORT TASK requests from each initiator. If the TST field equals 001b the target shall perform an action equivalent to receiving a series of ABORT TASK requests from only the requesting initiator.

All tasks <<, from all initiators,>> in the <<specified>> appropriate task set shall be aborted. The medium may have been altered by partially executed commands. All pending status and data for the appropriate task set <<that logical unit for all initiators>> shall be cleared.

No status shall be sent for any task. A unit attention condition shall be generated for all other initiators with aborted tasks (if any)<<in that task set>>. When reporting the unit attention condition the additional sense code shall be set to COMMANDS CLEARED BY ANOTHER INITIATOR.

Previously established conditions, including MODE SELECT parameters <<(see the SPC standard)>>, reservations, and contingent or auto contingent allegiance shall not be changed by the CLEAR TASK SET function.

((Was it really intended that the CLEAR TASK SET be able to come through a contingent allegiance regardless of initiator or is this in conflict with other sections?))

6.7 TERMINATE TASK

The effect of a TERMINATE TASK request on the task set depends on the task set error recovery option specified in the control mode page (see the SPC standard) and on whether or not a<<n>> contingent allegiance (NACA = 0) or auto contingent allegiance (NACA = 1) condition is generated.

IMPLEMENTORS NOTE: The TERMINATE TASK function provides a means for the initiator to request the logical unit to reduce the transfer length of the referenced command to the amount that has already been transferred. The initiator can use the sense data to determine the actual number of bytes or blocks that have been transferred. This function is normally used by the initiator to stop a lengthy read, write, or verify operation when a higher-priority command is available to be executed. It is up to the initiator to complete the terminated command at a later time, if required.

((Although not an integral part of this proposal, I thought the committee had agreed to obsolete TERMINATE TASK.))

7 Task Set Management

The rules for task set management only apply to a task after it has been entered into <<the>> a task set. A task shall be entered into <<the>> a task set unless a condition exists which causes that task to be completed with a status of BUSY, RESERVATION CONFLICT, TASK SET FULL, ACA ACTIVE or CHECK CONDITION (if caused by the detection of an overlapped command). A task may also be completed in this manner because of a CHECK CONDITION status caused by certain protocol-specific errors. In these cases, the task shall be completed as soon as the condition is detected.

((Although not related to this proposal, the last sentence above is very confusing which I presume will lead to interoperability issues. The author undoubtedly had specific conditions in mind not clear to the reader. All of the stated completions assume the logical unit receives a task. The first sentence implies that a task exists before it is entered into the task set. If the premise elsewhere that it does not exist until entered into the task set prevails then the last sentence would not be confusing.))

7.2 Task Management Events

The following is a description of 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 auto contingent allegiance condition has occurred for a task with NACA = 1.

CA: A contingent allegiance condition has occurred for a task with NACA = 0.

task abort: One of the events described in subclause 7.3 has occurred.

task completion: The device server has returned sent a service response of TASK COMPLETE for the task (see clause 5 and subclause 5.4).

task ended: A task has completed or aborted.

ACA cleared: An ACA condition has been cleared.

CA cleared: A CA condition has been cleared.

Subclause 7.4 describes the events, changes in task state and device server actions for a Simple, Ordered, ACA or Head of Queue task.

7.3 Task Abort Events
A Task Abort event is one of the following:
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 subclause 6.2;
c1) Completion of a CLEAR ALL TASK SETS task management function;
c) Completion of a CLEAR TASK SET task management function referencing the task set containing the specified task;
d) Completion of a PERSISTENT RESERVE with a Preempt and Clear action directed to the specified task;
e) A CA or an ACA condition was cleared and the QErr bit was set to one in the control mode page (see the SPC standard);
f) An ACA condition was cleared and the task had the ACA attribute;
g) A hard reset (see 5.6.6);
h) The return of an Execute Command service response of SERVICE DELIVERY OR TARGET FAILURE as described in clause 5.
i) A power on condition.

7.4 Task States

7.4.2 Blocked
A task in the Blocked state is prevented from completing due to contingent or an auto contingent allegiance condition. A task in this state shall not become a current task. While a task is in the Blocked state, any information the logical unit has or accepts for the task shall be suspended. If the TST field (See SPC-2) is 000b the blocked state is independent of the initiator. If the TST field equals 001b the blocked state applies only to the faulting initiator.

7.5.4 ACA Task ((Alternative 1))
A task having the ACA attribute shall be accepted into the task set in the Enabled state. As specified in 5.6.1.1, there <<may>> shall be no more than one ACA task per task set. **((It is not clear that this is specified in 5.6.1.1.))**

7.5.4 ACA Task ((Alternative 2))
A task having the ACA attribute shall be accepted into the task set in the Enabled state. <<As specified in 5.6.1.1,>> If the TST field equals 000b (See SPC-2) there <<may>> shall be no more than one ACA task per task set. If the TST bit equals 001b there may be more than one ACA task per LUN but there shall be no more than one ACA task per task set.

7.6.1 Transition S0:S1 (Ordered Task): Provided a CA or an ACA condition does not exist or if TST = 001b provided the task is not for the faulting initiator and the QErr field is not 01b (see SPC-2) , a dormant task having the ORDERED attribute shall enter the Enabled state when all older tasks have ended. If TST = 000b this transition shall not occur while a CA or an ACA condition is in effect for the task set.

7.6.2 Transition SO:S1 (Simple task): Provided a CA or an ACA condition does not exist or if TST = 001b provided the task is not for the faulting initiator, a dormant task having the SIMPLE attribute shall enter the Enabled state when all older Head of Queue and older Ordered tasks have ended. If TST = 000b this transition shall not occur while a CA or an ACA condition is in effect for the task set.

7.6.3 Transitions S0:S3, S2:S3: A task abort event shall cause the task to unconditionally enter the Ended state. **((Are there no CA or ACA exceptions?))**

7.6.4 Transition S1:S2: If TST = 000b a CA or an ACA condition shall cause an enabled task to enter the Blocked state. If TST = 001b a CA or an ACA condition shall cause an enabled task for the faulting initiator to enter the Blocked state.

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

7.6.6 Transition S2:S1: When a CA or an ACA condition is cleared and the QErr <<bit>> field is set to 00b in the control mode page (see the SPC-2 standard), a task in the Blocked state shall re-enter the Enabled state. When a CA or an ACA condition is cleared and the QErr field is set to 11b in the control mode page, a task in the Blocked state for other than the faulting initiator shall re-enter the Enabled state.

((The state diagrams in the pdf file are not easy to work with, so I have gone strictly by the descriptions. Consequently this needs to be checked against the state diagrams.))

((I elected to not propose what changes are needed for the 7.7 Examples and left that as a task (effort) to be accomplished after the requirements are agreed to.))

SAM Proposal Post Script:

5.6.1.2 Clearing an Auto Contingent Allegiance Condition
An auto contingent allegiance condition shall always be cleared after a power on condition or a hard reset (see 5.6.6). If the NACA bit is set to zero in the CDB control byte of the faulting command, then the SCSI-2 rules for clearing contingent allegiance shall apply. In this case, the logical unit shall also clear the associated auto contingent allegiance condition upon the return of sending sense data by means of the autosense mechanism described in 5.6.4.2.

While the SCSI-2 rules for clearing the ACA 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 6.3). **((Note that this requirement is in conflict with 5.6.1.1. Although not part of this proposal the choice of setting an error or ignoring the CLEAR ACA should be chosen by the committee rather than leaving the conflicting requirements.))**

5.6.4.1 Asynchronous Event Reporting

An error condition or unit attention condition shall be reported to a specific initiator once per occurrence of the event causing it. The logical unit may choose to use an asynchronous event report or to return CHECK CONDITION status on a subsequent command, but not both.

Notification of command-related error conditions shall be sent only to the device that initiated the affected task. **((Is this in conflict with the ACA requirements with TST = 000b?))**

Changes to SPI-2:

3.1.TBD contingent allegiance: One of the conditions of a task set following the return of a CHECK CONDITION or COMMAND TERMINATED status. A detailed definition of CA may be found in SCSI-2.

3.1.TBD task set: A group of tasks within a logical unit, whose interaction is dependent on the task management (queuing), CA, and ACA rules. <<defined in SAM >> See SAM.

3.1.19 exception condition: Any event that causes a SCSI device to enter an auto contingent allegiance (ACA) or contingent allegiance <<allegiant>> (CA) condition.

3.2 Symbols and abbreviations ((really acronyms))

ACA   Auto Contingent Allegiance (see 3.1.TBD)

CA   Contingent Allegiance (see 3.1.TBD and SCSI-2 as an informative reference)

11.1.1.1 Unexpected bus free

An unexpected bus free occurs when an initiator detects a BUS FREE phase (i.e., the release of BSY) that is not expected. Initiators only expect a BUS FREE phase to occur after one of the following occurs:

a) after a hard reset is detected;
b) after an ABORT TASK message is successfully received by a target;
c) after an ABORT TASK SET message is successfully received by a target;
d1) after a CLEAR ALL TASK SETS message is successfully received by a target;
d) after a<<n>> CLEAR TASK SET message is successfully received by a target;
e) after a<<n>> LOGICAL UNIT RESET message is successfully received by a target;
f) after a<<n>> TARGET RESET message is successfully received by a target;
g) after a<<n>> TERMINATE TASK message is successfully received by a target;
h) after a DISCONNECT message is successfully transmitted from a target;
i) after a TASK COMPLETE message is successfully transmitted from a target;
j) after a RELEASE RECOVERY message is successfully received by a target;

k) after the release of the SEL signal after a SELECTION or RESELECTION phase time-out;
l) after a transceiver mode change.

11.5.1.1 Message protocol rules

One or more messages may be sent during a single MESSAGE phase, but a message may not be split between multiple MESSAGE phases.

The first message sent by the initiator after a successful SELECTION phase shall be an IDENTIFY,
ABORT TASK SET, or TARGET RESET message. If a target receives any other message it shall switch to a BUS FREE phase (see 11.1.1.1). **((Why is it ABORT TASK SET and not ABORT TASK? Why not CLEAR TASK SET?))**

11.5.1.2 Message formats

One-byte, Two-byte, and Extended message formats are defined. The first byte of the message determines the format as defined in table 44.

Table 44 - Message format

Code	Message format
00h	One-byte message (TASK
01h	COMPLETE)
02h - <<13h>>14h	Extended messages
<<14h ->>15h	One-byte messages
16h - 17h	Reserved One-byte message<<s>>
18h - 1Fh	One-byte messages
20h - 24h	Reserved One-byte messages
25h - 2Fh	Two-byte messages
30h - 7Fh	Reserved Two-byte messages
80h - FFh	Reserved
	One-byte message (IDENTIFY)

11.5.4 Task management messages

Code	Support		Message Name		Direction	Negate ATN before last ACK
	Initiator	Target				
0Dh	Q	Q	ABORT TASK		Out	Yes
06h	O	M	ABORT TASK SET		Out	Yes
16h	O	O	CLEAR ACA		Out	Not required
14h	O	O	CLEAR ALL TASK SETS		Out	Yes
0Eh	Q	Q	CLEAR TASK SET		Out	Yes
17h	O	O	LOGICAL UNIT RESET (Note)		Out	Yes
0Ch	O	M	TARGET RESET		Out	Yes
11h	O	O	TERMINATE TASK		Out	Yes

Key: M=Mandatory support, O=Optional support, Q=Mandatory if tagged queuing is implemented
In=Target to initiator, Out=Initiator to target
Yes=Initiator shall negate ATN before last ACK of message.
Not required=Initiator may or may not negate ATN before last ACK of message (see 11.2.1).
n/a=Not applicable
***=Extended message

Note-The LOGICAL UNIT RESET message is mandatory if hierarchical addressing (see SCSI Controller Command Standard) is implemented by the target.

((Although not an integral part of this proposal, I thought the committee had agreed to obsolete TERMINATE TASK.))

11.5.4.2 ABORT TASK SET
The ABORT TASK SET message is defined in the SCSI-3 Architecture Model Standard.

In addition to the requirements in the SCSI-3 Architecture Model Standard the target shall go to the BUS FREE phase following the successful receipt of the ABORT TASK SET message.

If only an I_T nexus has been established, the target shall switch to a BUS FREE phase. No status or message shall be sent for the current task and no other task shall be affected.
The ABORT TASK SET message in the case of only an I_T nexus is useful to an initiator that cannot get an IDENTIFY message through to the target due to parity errors and just needs to end the current connection. No pending data, status, or tasks are affected. **((Although somewhat tangential to this proposal, I think the SCSI-2 authors got carried away including the note text from ABORT TASK into the body of ABORT TASK SET. Rather than being useful to use this message for parity errors it borders upon being dangerous. Any utility is better served by using the ABORT TASK message. I suggest that this paragraph be reduced to the first two sentences.))**

It is not an error to issue this message to an I_T_L nexus that does not have any pending or current tasks.

11.5.4.4 CLEAR ALL TASK SETS
The CLEAR ALL TASK SETS message is defined in the SCSI-3 Architecture Model Standard.
In addition to the requirements in the SCSI-3 Architecture Model Standard the target shall go to the BUS FREE phase following the successful receipt of the CLEAR ALL TASK SETS message.
((The CLEAR TASK SET message in SAM is defined to abort all tasks for all initiators in the specified task set. I believe the wording could have been construed to be the same as this function since no method was provided for identifying task sets other than by LUN number. However it seems more flexible to have two different messages to allow increased flexibility along with more expansive backward compatibility. But it may be adequate to define it such that ABORT TASK SET eliminates commands for a given initiator whether they be in one LUN s task set for all initiators or in one LUN s task set for just the current initiator with CLEAR TASK set clearing for all initiators regardless of how they are partitioned within the LUN.))

11.5.4.4 CLEAR TASK SET
The CLEAR TASK SET message is defined in the SCSI-3 Architecture Model Standard.
In addition to the requirements in the SCSI-3 Architecture Model Standard the target shall go to the BUS FREE phase following the successful receipt of the CLEAR TASK SET message.