From: Charles Monia
SAM Technical Editor

To: Members of X3T10

Subject: Proposed Responses to Comments from HP and IBM on SAM, Rev. 13

The following are proposed responses to the review comments received from Jeff Williams of Hewlett-Packard and George Penokie of IBM on revision 13 of SAM. Comments with no proposed response are accepted as written.

#001 (E) Section 1, Page 8, Paragraph 3

#SCSI-2 is listed as X3.131-1992. I believe it is actually X3.131-1994.

#002 (E) Section 1, Page 8, Paragraph 6

SCSI -2 should be SCSI-2.

#003 (E) Section 2.1.12, Page 9, Paragraph 1

The term being defined and it's use should be "completed command", not "command complete". This is consistent with terms like "aborted commands". It also removed the ambiguity between this term and "Command Complete".

>
> Comment Accepted.
>

#004 (E) Section 2.1.14, Page 9, Paragraph 1

This term should be "ended command", not "command ended".

Comment accepted.

#005 (E) Section 2.1.32, Page 10, Paragraph 1

This term should be "protocol indication", not "indication" in order to be consistent with terms like protocol service request, protocol service response and protocol service confirmation.

Comment accepted, with the understanding that the term should be "protocol service indication".

#006 (E) Section 2.1.58, Page 11, Paragraph 1

Why is the term "queue" defined? I thought we had purged the term from the document.

>
> Response:
>
The term is still in use -- see "Head of Queue" task attribute.

#007 (E) Section 2.1.61, Page 11, Paragraph 1

This term is, or should be, redundant with "protocol service request". In general, I noticed that request, indication, response, and confirmation are all listed twice, once alone and once with the "protocol service" added in front. I don't see the distinction between the two.

>

> Response:
>

The term "request" is a generic reference to the act of invoking a service. To make a "protocol service request" is to invoke a service provided by the protocol service layer. A task management request is a request directed to the task manager, etc.

#008 (E) Section 2.1.62, Page 11, Paragraph 1

This term is redundant with confirmed protocol service.
>
> Comment rejected.
>

A request-response transaction is a generic interaction between objects as specified in the definition.

#009 (E) Section 2.1.62, Page 11, Paragraph 1

This should be called "request-confirmation transaction".

>
> Comment rejected.
>

A remote procedure call is modeled as two-step request-response transaction. The Indication and Confirmation are interactions between the ULP and the protocol service layer.

#010 (E) Section 2.1.72, Page 12, Paragraph 1

This term is redundant with "device server".

>
> Comment rejected.
>

The term "server" refers to any object that provides services. I.e., the task manager is a server that performs task management services on behalf of initiators. The device server executes SCSI commands in response to device service requests, and so forth.

#011 (E) Section 2.1.74, Page 12, Paragraph 1

Remove, "...while in transit". There can be service delivery subsystem failure without the transaction ever being "in transit". For example, by differential transceivers may be disabled on a parallel bus when a single ended device is plugged into the bus. The data was never sent out of the local system. I think that removing this part of the sentence removes the requirement to define what "in transit" means in all cases.

>
> Comment rejected.
>

As specified in clause 3.5 on pp 31, from the viewpoint of the sender a transaction is defined as being "in-transit" as soon as it is passed to the service delivery interface for transmission. I believe that definition is consistent with your example.

#012 (T) Section 2.1.81, Page 12, Paragraph 1

Pending task is undefined, yet it is used in this definition.
>
> Comment accepted.
>

A definition for "pending task" will be added to the glossary.

#013 (E) Section 2.1.84, Page 13, Paragraph 1

"Task aborted" should be "Aborted Task". I also see this as a redundant definition of aborted task.

>
> Response:
>

The glossary term should be changed to "task abort".

#014 (E) Section 2.1.85, Page 13, Paragraph 1

"Task completed" should be "Completed Task". I also see this as a redundant definition of completed command.

>
> Response:
>

The glossary term should be changed to "task completion".

#015 (E) Section 2.1.86, Page 13, Paragraph 1

What is an ended task? Is it an aborted task, or a completed task? Is it redundant with other definitions.

>
> Response:
>

An ended task is a task that has completed or aborted.

#016 (E) Section 2.1.8x, Page 13, Paragraph 1

Task Management Indication and Task Management Confirmation are not defined.

>
> Response:
>

Indications and confirmations are only associated with interactions between the ULP and the protocol service layer. Task management requests and responses are ULP interactions between the application client and the task manager (which are conveyed by means of protocol service requests, protocol service indications, protocol service responses and protocol service confirmations).

#017 (E) Section 2.1.93, Page 13, Paragraph 1

Termination Pending should have (task state) listed after the term. It is one of the states in your task state machine definition.

>
> Comment accepted.
>

#018 (E) Section 2.4, Page 15, Paragraph 1

The sense key shall be either "INVALID FIELD IN CDB" or "INVALID FIELD IN PARAMETER LIST" depending upon where the invalid field is found. The statement, as it appears, would cause me to issue "INVALID FIELD IN CDB" even when there is an error in the parameter list of a command.

>
> Response.
>

This needs to be discussed by the working group. Apparently, there are a number of conflicting opinions on this issue.

#019 (E) Section 2.5.1, Page 15, Paragraph All

Clean up the format of the columns. They are hard to read.

#020 (E) Section 2.6, Page 18, Paragraph 2

"...([input1][,input2]...]||" should be "...([input1][,input2]...||".
Remove one "]" at the end of the input section.

#021 (E) Section 2.6, Page 18, Paragraph 10

The bullet item on the brackets should not be here. This is the same as the definition in the notation section (2.5.1). I assume that you could also include curly brackets in a procedure notation, but this is not included in the bullet list.

>
> Comment rejected,
>

In object notation, brackets "[...]" identify a group of objects, one of which is selected for inclusion in another object. In the procedure notation, brackets denote conditional or optional arguments and parameters.

#022 (E) Section 3.1, Page 22, Paragraph 4

The bullet should be "c." not "c)".

#023 (I) Section 3.1, Page 22, Paragraph 5

Last sentence. Everything humanly possible should be done to eliminate the use of internal behavior as a description of how something works. Every item in this standard should use externally observable behavior to describe the implementations. If internal behavior is required, then we have not done our jobs.

Remove this sentence and all internal behavior descriptions.

>
> Comment rejected,
>

The 'internal behavior' refers to behavior within some element of the model, such as a target device. SAM uses the "internals" of this hypothetical model, such as the task manager or device server, as the basis for describing the externally observable behavior that a real device should have.

#024 (E) Section 3.2, Page 23, Paragraph 1

Replace "request-response transaction" with "confirmed protocol service".

#025 (E) Section 3.2, Page 23, Paragraph 3

"..., the request becomes pending upon receipt and complete when the response is passed..." should be, "..., the request becomes pending upon receipt and completes when the response is passed...".

>
> Comment accepted.
>

#026 (E) Section 3.2.1, Page 25, Paragraph 1

"reading or writing to the media." should be "reading from or writing to the media.".

>
> Comment accepted.
>

#027 (E) Section 3.2.1, Page 26, Paragraph 2

First sentence should be removed. It is the definition of a task and is redundant with the definition in the definition section.

>
> Comment rejected.
>

A definition of the task object at this point is needed to show how it fits into in the context of the model.

#028 (T) Section 3.5.2, Page 31, Paragraph Last

Remove this paragraph. We state everything by arrival order so mentioning that we assume in order confuses and implies a requirement on the service delivery subsystem. Removing the paragraph does not change the meaning of arrival.

>
> Comment rejected.
>

This paragraph defines how the model deals with request-response ordering. The second sentence in that paragraph states:

"[The assumption of in-order delivery] is made to simplify the description of behavior and does not constitute a requirement."

#029 (E) Section 3.6, Page 32, Paragraph Fig 12

The service delivery interface is part of the device, but is described in the notation as being part of the service delivery subsystem. Either the figures need to change to reflect this, or the notation needs to move the SDI into the device rather than in the SDS.

>
> Comment rejected.
>

The service delivery interface physically resides in the device but is logically a part of the service delivery subsystem.

#030 (T) Section 3.6, Page 33, Paragraph 5

The target is listed as not being able to originate task management functions. What is the ruling then on the FCP target being able to issue an abort exchange? Is it legal according to SAM for a target to detect that an error has occurred and abort the task due to the detected error?

>
> Response:
>

A target-initiated "abort exchange" indicates that there is a problem which cannot be reported with a CHECK CONDITION status. I believe this kind of error is a "Service Delivery or Target Failure".

#031 (T) Section 3.6.1, Page 34, Paragraph 1

Since an initiator can have more than one initiator identifier, how does a target tell the difference between initiators? Is this going to be protocol specific, left undefined? In FCP for example, a login can tell the difference. In parallel it is more problematic since there is no way to tell if they are the same or if they are different.

>
> Response:
>

A target can't tell the difference and will implicitly consider two different initiator identifiers to represent two different physical devices.

#032 (T) Section 3.6.2, Page 35, Paragraph 10

There is no need for a base logical unit. It is no different than a logical unit zero. As a matter of fact they both have the same LUN.

>
> Comment accepted.
>

#032 (E) Section 3.6.2.1, Page 36, Paragraph 1

The task manager also controls the tasks based upon the ACA status, the tasks attributes (HEAD, ACA TAG, SIMPLE, ORDERED). It does not control them based solely upon the task management functions.

>
> Comment rejected.
>

The above attributes are associated with task set management. The rules for task set management are enforced by the device server.

#033 (T) Section 3.6.3, Page 36, Paragraph 5

The task set definition, as written, does not allow more than one untagged task in the task set at a time. I would suggest the following notation:

Task Set = 0{Tagged} + 0{Untagged}

>
> Comment accepted.
>

#034 (E) Section 3.6.3, Page 37, Paragraph 1

Remove the reference to queue. "Task Set (queue)" should be "Task Set".

>
> Comment accepted.
>

#035 (E) Section 3.6.3, Page 37, Paragraph 15 and 16

Tagged Task Address and Untagged Task Address are not very useful and cause confusion. You are implying that there is some special method used to refer to a task when using task management functions. There isn't. You refer to it in the same manner in which you do when you create the task. Eliminate these two items from your notation.

>
> Comment rejected.
>

The specification should distinguish between a task addresses that contains the tag component and one that doesn't.

#036 (E) Section 3.6.3, Page 38, Paragraph 4 and 5

"..protocol service request, The initiato r's", should be "...protocol service request, the initiator's...".

#037 (E) Section 3.7, Page 39, Paragraph 4

Change "...clause ?" to the correct reference.

#038 (E) Section 4, Page 42, Paragraph 1

"...sense data. returned..." should be "sense data returned..."

#039 (E) Section 4.1, Page 42, Paragraph All

The section 4.1 on page 42 is duplicated on top of page 43. Remove one of them.

#041 (T) Section 4.1, Page 43, Paragraph 3

Why is the statement allowed that says that media may be modified even if there is an invalid parameter or invalid field in a CDB? What cases would possibly allow you to modify media when the command is deemed to be incorrect?

>
> Comment accepted.
>

The sentence will incorporate the following SCSI-2 wording from clause 7.2, pp 79 of rev 10k.

"For all commands, if there is an invalid parameter in the command descriptor block then the target shall terminate the command without altering the medium."

#042 (T) Section 4.1.2, Page 44, Paragraph 2

The first sentence, as worded, says that the ACA will never be cleared. It should simply say that if the bit is a one, then the ACA shall be treated according 4.6.

>
> Comment accepted.
>

#043 (T) Section 4.1.2, Page 44, Paragraph 3

Why is ACA = 0 required? If I want to build a SCSI-3 device, you are requiring that I keep the old CA baggage even if I don't intend to operate with any SCSI-2 initiators. This is an unnecessary requirement which prevents SCSI-4 (yikes!) from eliminating support for CA entirely.

>
> Response:
>

Please bring the matter up for discussion during the working group.

#044 (E) Section 4.2, Page 46, Paragraph 2

Update the reference. "?" should be "5.6".

#045 (T) Section 4.3.2, Page 48, Paragraph

An indication and response are required. The model you present here does not follow the confirmed services model presented earlier.

>
> Comment rejected.
>

1. The only requirement for confirmed protocol services is the return of a confirmation. The model does not require a confirmed protocol service to generate an indication to or receive a response from the ULP. This is consistent with the protocol service interface defined for P1394 and the definition of a confirmed protocol service presented in clause 3.7.

2. The model assumes that the application client is unaware of the process of transferring data to or from it's data buffer. I believe this view reflects how initiators and host applications are implemented.

#046 (T) Section 4.3.3, Page 49, Paragraph

An indication and response are required. The model you present here does not follow the confirmed services model presented earlier.

>
> See response to item 045.
>

#047 (E) Section 4.5.1, Page 51, Paragraph Last

"initiato r's" should be "initiator's".

#048 (T) Section 4.5.2, Page 52, Paragraph

Considering the confusion over linked commands in the past, perhaps we should add in a statement that clarifies whether or not the linked command is an implied reservation or not. We have argued this one in committee a number of times and concluded only that it was not clear in the current documents. I suggest we try to add into the description the text to make it clear what we do in this case.

>
> Agree. Please raise this issue during the May plenary.
>

#049 (E) Section 4.6.1.1, Page 53, Paragraph Last

"mode byte" should be "control byte".

#050 (T) Section 4.6.1.1, Page 54, Paragraph 1

"The task shall then be entered into the task set if it meets all other conditions for acceptance." This does not convey that the task is the first one to be executed as was done in SCSI-2 CA. Stating that it is accepted is not sufficient. It must be stated that it is executed next and that it is untagged.

>
> Comment rejected.
>

1. According to the resolution of IBM comment 32 as recorded in X3T10/94-028 R0, page 33, it was agreed that any subsequent command, whether tagged or untagged, would cause the ACA condition to be cleared.

2. As show below, the text in SAM seems to reflect the behavior specified for SCSI-2.

The text you cite is as follows:

"If the ACA flag was set to zero in the faulting command, the auto contingent allegiance condition shall be unconditionally cleared upon receiving the next command from the faulted initiator as described in clause 4.6.1.2. The task shall then be entered into the task set if it meets all other conditions for acceptance."

The following is from section 7.6, , page 79, paragraph 1, sentence 3 of SCSI-2, rev 10k.

....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."

It's not clear how the version in SAM differs from the above with respect to the behavior of a subsequent command.

#051 (T) Section 4.6.1.2, Page 54, Paragraph 2

There should be no requirement for me to support the setting of the ACA bit to zero.

>
> See response to item 043.
>

#052 (E) Section 4.6.4, Page 58, Paragraph 3

"initiato r's" should be "initiator's".

#053 (T) Section 5, Page 60, Paragraph 12

Abort Task should not be required. It is only required if tagged tasks are supported. Abort Task Set makes more sense to be required than does Abort Task

>
> Comment rejected.
>

While a case can be made for either approach, there seems to be no compelling reason to change this yet again. It is more important to begin stabilizing the document.

#054 (E) Section 5.1, Page 61, Paragraph 6

"A response of Function Complete indicates that the task is not in the task set." implies that there is some other response sent if the task was aborted. This should say that the function complete is sent once the task is aborted. Also, the task not being present is not an error.

>
> Comment accepted.
>

#055 (E) Section 5.4, Page 63, Paragraph 1

"... in the specified task set shall be ." Oops, shall be what? Aborted.

>
> Yes.
>

#056 (T) Section 5.4, Page 63, Paragraph 1

"All data for all terminated tasks shall be cleared." Two things. First, it should be aborted tasks not terminated. Second, the data cleared is the sense data. This implies that I must flush my buffers for any data that these tasks may use. This is not the intent of the statement.
>
> Response.
>

1. "Terminated' will be replaced with 'aborted'.

2. The corresponding wording in the SCSI-2 spec. (rev 10k, section 6.6.4, pp 57, first paragraph) says:

"....All pending status and data for that logical unit or target routine for all initiators shall be cleared."

While the wording in SAM should be changed to agree with the above, it's not obvious that sense data is included. This item should be discussed at the May working group.

#057 (T) Section 6.1, Page 67, Paragraph 5

Your current task definition does not include a task which is sending status. Current does not mean only data, it includes status or any other information transfer.

>
> Comment accepted.
>

#058 (E) Section 6.2, Page 67, Paragraph 2 and 3

"...that were in the task set before the referenced task." should be, "...that were entered into the task set before the referenced task was entered into the task set.". Just saying before the referenced task leaves the ambiguity that they could be before the referenced task in the task set (i.e. HEAD).

>
> Comment accepted.
>

Same comment for paragraph 3.

>
> Comment accepted.
>

#059 (E) Section 6.2, Page 68, Paragraph 2

4.6.2 is not a correct reference. 6.2.1 is the correct reference.

>
> Comment accepted.
>

#060 (E) Section 6.2, Page 68, Paragraph 3

"...or is unable to continue command execution because of task termination" does not correlate with this state. Task termination is a very orderly procedure. This implies that the task can't be completed when it can, and is, completed in a predefined fashion.

>
> Response:
>

Task completion is any event that culminates in the return of a Command Complete response from the device server, including successful command execution. Task termination is one such event.

#061 (E) Section 6.2.1, Page 68, Paragraph 3

This should also state that the ABORT TASK SET must also reference the initiator identifier. That is, an ABORT TASK SET from initiator 1 does not effect the tasks of initiator 2.

>
> Comment accepted.
>

#062 (T) Section 6.2.1, Page 68, Paragraph 5

The occurrence of the ACA condition with QErr set does not effect the task set. The CLEARING of the ACA condition when the QErr bit is set causes the tasks to be cleared.

>
> Comment accepted.
>

#063 (I) Section 6.2.1, Page 68, Paragraph All

This list groups a lot of items into the category of abort. Many of these have no correlation to an abort as it is known today. For example, if I am reading from the media and get an Abort Task for a command not "internally active", I just delete it and continue with the current task. If I get a reset, I blow everything away, including the read from the disk. These are very different, yet you are grouping them into the same term.

>
> Response:
>

The paragraph will be reworded to clarify that the task abort events listed are relative to a specific task. The state diagrams show how the state of that task changes in response to these events.

#064 (T) Section 6.3, Page 70, Paragraph All

This entire section is extremely confusing. It is making assumptions about internal states of the device and is not a model based upon the externally observable behavior of the device. When we started work on the queuing model, we assumed that the internal states of a device were out of bounds for discussion. This model has put everything back into the internals of a device. I cannot vote to accept any model which is "device-centric" rather than "bus-centric".

>
> Response:
>

1. The use of a state machine is an accepted way to rigorously specify how externally observable inputs are transformed into externally observable outputs (AKA: observable device behavior). The method has the proven advantage of exposing incomplete or incorrectly specified behavior.

2. A state machine model involves no assumption about the internal behavior and states of a real device. The only requirement is that the real device, however it is implemented, faithfully emulate the external behavior manifested by the state machine.

3. If, by bus-centric, you mean a description of behavior using events observed on the bus, I believe SAM meets the intent of that criterion.

Comment:

1. Page 9 Section 2.1.23 - When does execution start? (e.g. On a write command some or all of the data is moved across the interface. Has the write command started execution or not?)

>
> From SAM's point of view, execution starts when the task enters the Enabled state.
>

As stated in clause 6, task behavior in SAM is based on behavior visible to the application client. (e.g., the device driver). That behavior in SAM excludes movement of data across the interface.

Clause 6, second paragraph, third sentence states:

"To define these and other aspects of behavior, SCSI-3 protocol and interconnect standards may impose other requirements, outside the scope of this standard, that are related to observable behavior within the protocol or interconnect layers."

i.e., While the movement of data across the interface is not related to task execution as far as SAM is concerned, protocol specifications may impose requirements to regulate such behavior.

Comment:

2. Page 11 - Pending Task should be defined in the glossary

>
> Comment accepted.
>

---

3. Page 15-16 - Several of the entries in the symbols list do not have any spaces between the second column and the third column.

4. Page 39 Section 3.7 1st paragraph after figure 17 near the end of the paragraph there is an undefined cross-reference.

5. Page 42 - The heading 4.1 and the two lines following it are duplicated on the top of page 43.

---

6. Page 44 - 1st paragraph after table 3 - The second to the last sentence should have a cross-reference to clause 6.6.1.

>
> Comment accepted (with the understanding that the suggested reference is to clause 4.6.1).
>

7.*Page 44 - 1st paragraph after table 3 - This is the start of the ACA description problems. There should be a statement in this paragraph that says that if the ACA bit is zero the SCSI-2 rules for handing exception conditions shall be used by the Target. No further definition is required and in fact the definitions in clause 4.6.1 only confuse anyone who has implemented CA in SCSI-2.

>
> Comment Rejected.
>

This contradicts the working group consensus reached during the January meeting which stated that the desired behavior must be explicitly defined in SAM. (See response to IBM comment 32 on page 33 of X3T10/94-028R0 . This document is included in X3T10 1994 mailing 1).

Another problem is that the logical unit's response to a new ACA condition is determined on a command by command basis. Simply referring to SCSI-2 behavior is insufficient for a logical unit that supports ACA bit values of one or zero in the CDB control byte.

The proper way to resolve this issue is to identify and correct specific problems in SAM.

8. Page 46 - section 4.2 - command terminated paragraph - There is a undefined cross-reference at the end of the first sentence.

9. Page 46 and other places throughout the document - Statuses and messages have been changed from 'Queue' to 'Task Set'. Was this change agreed to by the committee? If so OK if not it should be voted on.

>
> Comment rejected.
>

1. There are no messages defined in SAM. You seem to be equating the task management function names in section 5 with message names in SIP.

2. The use of "task set" instead of "queue" throughout SAM was at the request of reviewers (including IBM), who felt that the task set concept more accurately reflected the new queuing model. The working group consensus reached in January reaffirmed that decision (see X3T10/94-028R0 , response to item 39 on page 35).

3. The document review and letter ballot process now underway is the way in which the committee membership at large may review and vote on such issues.

10. Page 50 list under entry b - There should be an 'or' between RESET,

TARGET RESET.

11. Page 53 section 4.6.1.1 last paragraph - The first sentence should be changed from 'the mode byte of the' to 'the control byte of the'.

==========

12. *Page 53 section 4.6.1.1 last paragraph - The statement 'shall be unconditionally cleared upon receiving the next command from the faulted initiator' is not the behavior described in SCSI-2 for this condition. (See comment number 7 for the solution.)

>
> Comment rejected:
>
1. See response to item 7.

2. The full text of the sentence you quote is:

"If the ACA flag was set to zero in the faulting command, the auto contingent allegiance condition shall be unconditionally cleared upon receiving the next command from the faulted initiator as described in clause 4.6.1.2."

The following is taken from the SCSI-2 standard (pp 79, rev 10K)

"7.6 Contingent Allegiance

The contingent allegiance condition shall exist following the return of CHECK CONDITION or COMMAND TERMINATED status. The contingent allegiance condition shall be preserved 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...."

From the above, it is not clear how the SAM requirement you cite is at variance with the SCSI-2 definition.

==========

13. *Page 54 section 4.6.1.1 first paragraph last sentence - What are the conditions for 'acceptance'? Where in the task set is the command placed; is the task treated as a Head of Queue, Simple, or Ordered task?? (See comment number 7 for the solution)

>
> Response:
>

1. A task is acceptable unless:

a) The logical unit detects an overlapped command condition.
b) There is a reservation conflict,
c) The logical unit discovers that a field in the CDB is set to an unacceptable value.
d) There is a queue full condition,
e) There is a busy condition,

f) The task is ineligible for inclusion in the task set due to an ACA condition.

2. The task's placement in the task set is based on the task attribute specified by the application client..

---

14. *Page 54 section 4.6.1.1 2nd paragraph first sentence states 'The completion of the new task with'. I do not know what is meant by 'new task' in that sentence. I assume it is an attempt to reword the 2nd paragraph in section 2.1.1 of the SCSI-3 Queuing Model but the message seems to have been lost.

>
> Response:
>

"The new task" refers to the task discussed in the two paragraphs of 4.6.1.1 before the one you cite -- i.e.. a task created while an ACA condition exists. It expands on section 2.1.1 in the appendix to address the issue of how the ACA bit in a new task is to be handled after an auto contingent allegiance condition is in effect.

The use of the term "new task" seems in keeping with annex A usage. As an example, the following is from annex A, section 6.

"If simple or ordered tasks are accepted into the task set after an Ordered task the target shall suspend any information accepted for the new task ...."

---

15. *Page 54 section 4.6.1.1 3rd paragraph first sentence should be changed to 'faulting command, then the auto contingent allegiance condition shall not be cleared and a new task shall be entered into the..'

>
> Comment rejected.
>

The last sentence of the paragraph you cite describes the required behavior.

---

16. *Page 54 section 4.6.1.2 3rd paragraph last sentence: I have not idea what this sentence means.

>
> Response::
>

The paragraph you refer to states:

"The state of all tasks in the task set when an auto contingent allegiance condition is cleared shall be modified as described in clause 6. A task having the ACA attribute shall be aborted".

In the context of the complete paragraph, the meaning of the last sentence seems clear.

---

17. *Section 4.6.1: After careful study of this section there seems to be several concepts defined in the SCSI-3 Queuing Model that are not here. The Missing concepts are list below:

2.1.2 Response to Auto Contingent Allegiance Condition

If a Task becomes a current task because of a previous request for information that information shall be suspended until the ACA is cleared.

>
> Comment rejected.
>

The requirements you mention pertain to tasks in the Held state. Regarding such tasks, the second paragraph of clause 6.3 on pp 71 states:

"While an Ordered task is in the Held state, any information the logical unit has or accepts for the Ordered task shall be suspended. While a Simple task is in the Held state, any information accepted on behalf of the Simple task shall be suspended."

This paragraph seems to contain the requirement you mention above.

2.1.3 Auto Contingent Allegiance Processing

All SCSI operations are permitted while processing an ACA Task.

>
> Comment rejected.
>

It is not clear what you mean by "SCSI operations". If you are referring to task management functions, then this concept is already included in SAM.

2.1.4 Clear Auto Contingent Allegiance Task Management Function

The target shall clear the Auto Contingent Allegiance and complete the current Task on acceptance of this task management function.

If the target accepts a Clear Auto Contingent Allegiance Task Management Function and no Auto Contingent Allegiance Condition is in effect for that initiator on that task set, then the target shall complete the current Task.

>
> Comment rejected.
>

The technical requirements in the text you cite are specified in clause 5.3 of SAM.

This case is one example of the problem in literally copying the queuing model text word-for-word to SAM.

In SAM, a task represents an SCSI command (or a string of linked commands). So, "clearing the current task", if taken literally, might imply that some pending command was to be aborted.

Now consider the corresponding text from revision 3 of the queuing model (dated 29 September, 1992):

"I If the target receives a Clear Auto Contingent Allegiance
| Condition message when no Auto Contingent Allegiance Condition
| exists for that I_T_x_y nexus the target shall immediately complete
| the current I/O Process."

In this version, it seemed like the intended behavior was to treat the CLEAR ACA request as a NOP if there was no ACA. This makes perfect sense. Comparing the rev 3 text with the rev 6 wording, it appeared as if revision 6 incorrectly assumed that a "task" is SAM was synonymous with a SIP "I/O process". The text in SAM was an attempt to reflect the intent as specified in the rev 3 document.

==========

If a Clear Auto Contingent Allegiance Task Management Function occurs when an ACA Task is pending then the ACA Task shall be aborted and the auto contingent allegiance shall be cleared.

>
> Comment rejected.
>

The specified behavior is shown in the state diagram of figure 26, the state table of figure 27 and described in section 6.4.1.1.

==========

18.* Page 54 section 4.6.2 last paragraph 2nd sentence should be changed to '...tasks for the initiator that caused the overlapped commands in the task set...'.

>
> Comment accepted.
>

==========

19. section 4.6.2 - The list of things that can occur to free up tags is not listed. The list out of the SCSI-3 Queuing Model follows:

2.2 Duplicate Tag Handling

When issuing a tagged task the initiator shall not reuse the tag to create a new task until:

-A service response of Command Complete is received with a status other than INTERMEDIATE or INTERMEDIATE-CONDITION MET.
-A service response of Service Delivery or Target Failure is received. In this case, system implementations shall guarantee that the task associated with that command has been terminated.
-A power on condition occurs.
-A Target Reset Task Management request occurs.
-An Abort Task Management request occurs.
-An Abort Task Set Management request occurs.
-A Clear Task Set Management request occurs.
-A unit attention of TASKS CLEARED BY ANOTHER INITIATOR is reported.
-A unit attention of POWER ON, RESET or TARGET RESET is reported.

>
> Comment rejected.
>

The above list is included in clause 4.4 of SAM.

20.* Page 58 section 4.6.5 - In this section it must me made clear that the clearing of the unit attention condition does not automatically clear the auto contingent allegiance condition if the ACA bit is set to one.

>
> Comment accepted.
>

Of course the above condition only applies when the unit attention is reported by returning a CHECK CONDITION status.

---

21. Page 62 section 5.3 last paragraph - The last sentence should be changed to ' ...subject to the task set management...' and the wrong clause is referenced; it should reference clause 6.

>
> Comment accepted.
>

---

22. *Page 67-74 sections 6.2 - 6.4.4.1 - I attempted to understand this section to see if it matched the SCSI-3 Queuing Model but I could not. The problems I had were:
-Cross-references in almost every sentence.
-Figure 23 is incomprehensible to me (Bring back the bubbles).
-The words previous and prior drive me crazy what's wrong with simple words like before and after.
-Sections 6.3.1 to 6.3.8.1 are frustrating trying to understand.

To net it out: I cannot determine if SAM complies with the SCSI-3 Queuing Model. The only thing I can go by is section 6.5 which contains the task management examples. Everything in the examples looks correct except for one minor editorial change (see below). But those are only examples and I cannot assume the other sections correctly define the actions within the examples.

>
> Comment rejected.
>

1. I believe SAM is in full compliance with the queuing model.

2. Given the generality of the remarks and the failure to identify a specific technical error, there appears to be no basis for changing the document.

---

23. Page 76 figure 29 - The lower left task set brackets for the enabled tasks should be extended to include the Head of Queue (Task 7).