Memo to: ANSI X3T9.2

Memo from: Attendees of first meeting,
Committee on Command Queuing


| | |
|---|---|
| James McGrath<br>Quantum<br>1804 McCarthy Blvd<br>Milpitas, CA 95035 | Jeff Stai<br>Western Digital<br>2445 McCabe Way<br>Irvine, CA 92714 |
| Robert N. Snively<br>Adaptec<br>580 Cottonwood<br>Milpitas, CA 95035 | I. Dal Allan<br>ENDL<br>14426 Black Walnut Ct.<br>Saratoga, CA 95070 |


Date: April 3, 1987

Subject: Command Queueing Proposal for SCSI-2 (Rev 1.0)

The SCSI architecturally supports very complex multi-tasking environments. The architecture already supports the queueing of commands from different initiators against a single SCSI device and LU. This is referred to in this proposal as untagged queuing. With minor additions, the SCSI-2 is also capable of supporting the queueing of commands from the same initiator against a single SCSI device and LU. This is referred to in this proposal as tagged queuing. Note that tagged queuing is a superset of untagged queuing.

This document defines the extensions that are required to provide both forms of queuing. It is a working paper of the committee on Command Queuing, and in this version combines elements of previous proposals from Quantum (X3T9.2/87-49), Emulex (X3T9.2/87-46), Control Data (X3T9.2/86-82), and Adaptec (X3T9.2/87-54) on this topic.

This letter briefly describes the concept intended to be implemented by the attached text revisions. Untagged queueing is managed the same way as in the SCSI X3.131-1986. All commands are untagged and no overlapped commands for a given LUN from a given initiator are allowed.

Tagged queueing is a far more powerful concept. An identified command may be queued according to the specified rules from any intiator to any LUN.

Conceptually, each command transmitted to a target is transmitted either with an unordered queue tag, with an ordered queue tag, with a head of queue tag, or without a queue tag. If the command is transmitted to the target without a tag, it indicates that the command must be executed in the order received from the



transmitting initiator. No time relationship between the execution of untagged commands from other initiators is guaranteed except through use of the reservation instructions. Only one untagged command can be pending from a particular initiator for each LUN. If the command is transmitted to the target with an unordered queue tag, the command can be executed in any order on the selected LUN with respect to other unordered tagged commands from any initiator. If the command is transmitted to the target with an ordered queue tag, the command must be placed in the queue after all unordered queue tagged commands previously received from any initiator and before any unordered queue tagged commands received later. Ordered queue tagged commands are executed in the strict order received, regardless of initiator. In a stream of tagged commands, the order of execution of an untagged command with respect to various types of tagged commands is not defined. Head of queue tagged commands are placed in the queue for execution before all other commands. Any commands that may have been taken from the queue and for which execution has been started will not be pre-empted by head of queue tagged commands. Sequential head of queue tagged commands are executed in FIFO order. [THERE IS SOME CONCERN THAT LIFO MAY BE THE RIGHT ORDER.]

The use of the UNORDERED QUEUE TAG implies that the command or link of commands may be queued by the target SCSI device and executed against the LU in an arbitrary order. The transmission of a command with the ORDERED QUEUE TAG message implies that that tagged ordered command or link of commands must be executed in the strict order received by that LUN regardless of the initiator.

The use of the ORDERED QUEUE TAG implies that the command or link of commands must be queued by the target SCSI device and executed against the LU in the strict order received. All previous UNORDERED QUEUE TAG labeled commands must be executed before the command labeled by the ORDERED QUEUE TAG. All subsequent UNORDERED QUEUE TAG labeled commands must be executed after the command labeled by the ORDERED QUEUE TAG, regardless of initiator. The transmission of a command without any tagging message implies that the untagged command or link of commands must be executed in the order transmitted relative to other commands to that LUN from that initiator.

The use of the HEAD OF QUEUE TAG implies that the command must be queued by the target SCSI device and placed first in the queue for execution immediately after termination of any command that may be active. No active command is interrupted or terminated by a HEAD OF QUEUE TAG tagged command. Consecutive head of queue tagged commands are executed in FIFO order. **[LIFO order? ]**

Only one command with a given tag value can be active against an LU from an initiator at a time. The number of queue tag bits assures that an unused tag can easily be found.
If the command queue is full, an indication must be given to an initiator that no more commands can be enqueued until one or more

of the already queued commands is complete. Execution of the command is halted and must be reinitiated by the initiator. The full queue message may be presented before command transfer, interrupting command transfer, or immediately after command transfer. No command execution can take place. The SCSI goes to bus free state after the full queue message has been accepted by the initiator.

If the command queue is full and an untagged command is received, BUSY status is presented and the command is not executed or queued. The initiator must reinitiate the command at a later time.

The command queue tagging protocol is chosen to be compatible with SCSI X3.131-1986 as well as functional with the SCSI-2 standard. The command queue tagging protocol is further chosen to allow disconnection any time after the IDENTIFY message and the QUEUE TAG message have been transmitted.

Both deferred errors and normal errors are handled in the normal manner. Normal errors are indicated by CHECK CONDITION status. REQUEST SENSE commands are executed against the same queue tag to obtain the information stored for the failing operation. REQUEST SENSE commands are normally labeled as head of queue tagged commands to encourage immediate execution. Long delays in executing REQUEST SENSE may cause some sense information to be discarded if a number of errors occur, causing the sense information buffer capacity to be exceeded. Deferred errors are normally related to a command that has long since completed. As such, there is no attempt to point back to the queue tag assigned to the original failing command. Since command queueing with disconnection performs much the same function as the Immediate bit, it is expected that few fully queued commands will choose to use the Immediate bit. This limits the set of errors that can become deferred errors.

The proposal is added to the end of section 6 of the SCSI-2 document. The section will include both a description of untagged command queueing and tagged command queueing. In addition to the inclusion of section 6, section 5 will pick up new messages in table 5-4 and in section 5.5.



To add the extra extended messages required to implement this proposal, Table 5-4 is replaced with the following table:

Table 5-4: Extended Message Codes

| Code (y) | Description |
|---|---|
| 00h | MODIFY DATA POINTER (Optional) |
| 01h | SYNCHRONOUS DATA TRANSFER REQUEST (Optional) |
| 02h | EXTENDED IDENTIFY (Optional) |
| 03h | UNORDERED QUEUE TAG (Optional) |
| 04h | HEAD OF QUEUE TAG (Optional) |
| 05h | ORDERED QUEUE TAG (Optional) |
| 06h | Reserved |
| 07h | QUEUE FULL (Optional) |
| 08h | ABORT TAG (Optional) |
| 09h | CLEAR QUEUE (Optional) |

The new messages are specified by the addition of the following text to section 5.5:

### 5.5.7 UNORDERED QUEUE TAG (Optional)

Table 5-8: UNORDERED QUEUE TAG

| Byte | Value | Description |
|---|---|---|
| 0 | 01h | Extended message |
| 1 | 02h | Extended message length |
| 2 | 03h | UNORDERED QUEUE TAG |
| 3 | x | Tag value |

The UNORDERED QUEUE TAG message (Table 5-8) shall be implemented if tagged queuing is implemented. It is transmitted by both the target and the initiator. It is transmitted immediately after the IDENTIFY message to provide a tag for that particular command or link of commands during the entire period that the command is queued or being executed. The tag consists of the quadruple of the Initiator, Target, LUN, and Tag Value. A command tagged by the UNORDERED QUEUE TAG message will be enqueued for execution by the target in a target unique manner.

The UNORDERED QUEUE TAG message out phase is requested by the initiator immediately after the IDENTIFY message by leaving ATN

active after the time that ACK drops in response to the REQ for the IDENTIFY message. Target devices supporting tagged queueing will not disconnect until after both the IDENTIFY and the UNORDERED QUEUE TAG messages out have been received.

The UNORDERED QUEUE TAG message in is provided immediately after the reselection IDENTIFY message in has been presented to complete the identification process of the operation being continued. If no type of tagging message in is provided, it is assumed that the reconnection is being made on behalf of an active untagged command.

If a SCSI device receives an UNORDERED QUEUE TAG message and does not support tagged queueing, the device will respond with a MESSAGE REJECT message and complete the command as if it were an untagged command. If a target receives a message with a Tag Value that is currently being used for that Initiator/LU, then it will respond as if an untagged overlapped command had occurred. Both the original command and the overlapping command will be aborted, regardless of whether the original command is enqueued for future execution or is actively being executed. (See section 6.5.1) [See proposal X3T9.2/87-51] If an initiator receives a message with a Tag Value that is not currently being used for that Target/LU combination, then it will respond with an ABORT message [special message?].

### 5.5.8 HEAD OF QUEUE TAG (Optional)

Table 5-9: HEAD OF QUEUE TAG

| Byte | Value | Description |
|---|---|---|
| 0 | 01h | Extended message |
| 1 | 02h | Extended message length |
| 2 | 04h | HEAD OF QUEUE TAG |
| 3 | 00h | Tag Value |

The HEAD OF QUEUE TAG message (Table 5-9) shall be implemented if tagged queuing is implemented. It is sent by both the target and the initiator. It is transmitted immediately after the IDENTIFY message to provide a tag for that particular command or link of commands as it is being queued or executed by the initiator and the LU. The HEAD OF QUEUE TAG message requests that the tagged command be placed first in the queue for execution. The HEAD OF QUEUE tagged command will not pre-empt commands already dispatched from the queue, but will be executed next after a currently executing command. Subsequent HEAD OF QUEUE tagged commands will be inserted at the head of the queue for execution in FIFO [ LIFO ?] order.

The Tag value is used in conjunction with the Initiator, Target, and LU number to generate a quadruple that uniquely identifies



the associated command.

The HEAD OF QUEUE TAG message out phase is requested by the initiator immediately after the IDENTIFY message by leaving ATN active after the time that ACK drops in response to the REQ for the IDENTIFY message. Target devices supporting tagged queueing will not disconnect until after both the IDENTIFY and the HEAD OF QUEUE TAG messages out have been received.

The HEAD OF QUEUE TAG message in is provided immediately after the reselection IDENTIFY message in has been presented to complete the identification process of the operation being continued. If no type of tagging message in is provided, it is assumed that the reconnection is being made on behalf of an active untagged command.

If a SCSI device receives a HEAD OF QUEUE TAG message and does not support it, the device will respond with a MESSAGE REJECT message and complete the command as if it were an untagged command. If a target receives a message with a Tag Value that is currently being used for that Initiator/LU, then it will respond as if an untagged overlapped command had occurred. Both the original command and the overlapping command will be aborted, regardless of whether the original command is enqueued for future execution or is actively being executed. (See section 6.5.1) [See proposal X3T9.2/87-51] If an initiator receives a message with a Tag Value that is not currently being used for that Target/LU combination, then it will respond with an ABORT message [special message?].

### 5.5.9 ORDERED QUEUE TAG (Optional)

Table 5-10: ORDERED QUEUE TAG

| Byte | Value | Description |
|---|---|---|
| 0 | 01h | Extended message |
| 1 | 02h | Extended message length |
| 2 | 05h | ORDERED QUEUE TAG |
| 3 | x | Tag value |

The ORDERED QUEUE TAG message (Table 5-10) shall be implemented if tagged queueing is implemented. The message is sent by both the target and the initiator. It is transmitted immediately after the IDENTIFY message to provide a logical identifier for that particular command or link of commands as it is being executed by the initiator and the LU. The ORDERED QUEUE TAG message requests that the labeled command be placed in the queue

for execution in the strict order received. All earlier commands of any tag type are executed before the ORDERED QUEUE TAG labeled command while all commands transmitted later must be executed after the ORDERED QUEUE TAG labeled command.

The Tag value is used in conjunction with the Initiator, Target, and LU number to generate a quadruple that uniquely identifies the associated command.

The ORDERED QUEUE TAG message out phase is requested by the initiator immediately after the IDENTIFY message by leaving ATN active after the time that ACK drops in response to the REQ for the IDENTIFY message. Target devices supporting tagged queueing will not disconnect until after both the IDENTIFY and the ORDERED QUEUE TAG messages out have been received.

The ORDERED QUEUE TAG message in is provided immediately after the reselection IDENTIFY message in has been presented to complete the identification process of the operation being continued. If no type of tagging message in is provided, it is assumed that the reconnection is being made on behalf of an active untagged command.

If a SCSI device receives a ORDERED QUEUE TAG message and does not support it, the device will respond with a MESSAGE REJECT message and complete the command as if it were an untagged command. If a target receives a message with a Tag Value that is currently being used for that Initiator/LU, then it will respond as if an untagged overlapped command had occurred. Both the original command and the overlapping command will be aborted, regardless of whether the original command is enqueued for future execution or is actively being executed. (See section 6.5.1) [See proposal X3T9.2/87-51] If an initiator receives a message with a Tag Value that is not currently being used for that Target/LU combination, then it will respond with an ABORT message [special message?].

32

### 5.5.10 QUEUE FULL (Optional)

Table 5-11: QUEUE FULL

| Byte | Value | Description |
|---|---|---|
| 0 | 01h | Extended message |
| 1 | 02h | Extended message length |
| 2 | 07h | QUEUE FULL |
| 3 | 00h | Tag Value |



The QUEUE FULL message (Table 5-12) shall be implemented if tagged queueing is implemented. The message may be sent by the target. It is transmitted when an UNORDERED QUEUE TAG, ORDERED QUEUE TAG, or HEAD OF QUEUE TAG message is received from the initiator and when no commands can be accepted from that initiator for that LUN. The Tag Value contains the value from the received tagging message as a verification of the command which will not be executed. The command is not executed and must be reinitiated at a later time. The target goes to the BUS FREE phase after transmission of the QUEUE FULL message is successfully completed. Completion of the message transfer is defined as the successful transmission of the four bytes of the message, the last byte being correctly acknowledged with the ACK signal without the ATN signal being transmitted.

**[ We should probably define a minimum time between the last fall of REQ and the establishment of BUS FREE. ]**

**[This message must be added to the list of "expected" BUS FREE conditions when the wording is finalized. See proposal X3T9.2/87-55.]**

### 5.5.11 ABORT TAG (Optional)

Table 5-12: ABORT TAG

| Byte | Value | Description |
|---|---|---|
| 0 | 01h | Extended message |
| 1 | 02h | Extended message length |
| 2 | 08h | ABORT TAG |
| 3 | x | Tag value |

The ABORT TAG message (Table 5-13) shall be implemented if tagged queueing is implemented. The message may be sent by the initiator. When this message is received, the target will dequeue the command identified by the tag value. If the target has already started execution of the command, the execution will be halted. The media contents may have been modified before the execution was halted. In either case, any pending status and/or data for that command is cleared. No status or ending message will be sent for the identified command. Pending status, data, and commands for other queued operations will not be affected. If the target is currently connected for the command specified in the tag value, the target will go to BUS FREE phase. Execution of other commands queued against the specified LU will continue in the normal manner.

### 5.5.12 CLEAR QUEUE (Optional)

Table 5-13: CLEAR QUEUE

| Byte | Value | Description |
|---|---|---|
| 0 | 01h | Extended message |
| 1 | 02h | Extended message length |
| 2 | 09h | CLEAR QUEUE |
| 3 | x | Reserved |

The CLEAR QUEUE message (Table 5-14) shall be implemented if tagged queueing is implemented. The message may be sent by the initiator. When this message is received the target will perform an action equivalent to receiving a series of ABORT TAG messages for all commands currently queued against the identified LU for the requesting initiator. All commands in the queue are cleared from the queue. All executing commands are immediately halted. The media may have been modified by partially executed commands. All pending status and/or data for that LUN for commands from the requesting initiator are cleared. No status or ending message will be sent for any of the outstanding commands. If the target is currently connected for the command specified in the tag value, the target will go to BUS FREE phase. Any commands in execution or in the queue for that LU from other initiators will be executed in the normal manner.

Note that previously established conditions, including MODE SELECT parameters and RESERVE/RELEASE conditions are not changed by the CLEAR QUEUE command.

Section 6.6 will describe the queueing capability of the SCSI with the text provided below. Section 6.6.3 demonstrating examples of queueing may be appropriately moved to an appendix.

## 6.6 Command Queueing

Command queueing allows a target to accept more than one command at a time for execution by a particular LU. Untagged command queueing allows an LU to simultaneously accept commands from more than one initiator at a time. Tagged command queueing allows an LU to simultaneously accept multiple commands from each initiator to each LU. By using command queueing, a target avoids the overhead of presenting BUSY status to a command when it is actually busy processing other commands. In addition, command transmission overheads are decreased because command transmission can be overlapped with mechanical delays in the target LU. Within the specified limitations, the commands may be dequeued, initiated, and completed in an arbitrary order established by the target.



### 6.6.1 Untagged Command Queueing

Untagged command queueing allows an LU to accept a new command from an initiator while the LU is actually processing commands for another initiator. Only one command may be active for each LU from each initiator at any time.

A new command may be accepted for the LU at any time that the SCSI bus is free whether or not another command from a different initiator is being executed by the LU. The target may force the command to disconnect before or after the Command Descriptor Block has been received. It is preferable for the target to accept the Command Descriptor Block so that processing may begin immediately upon termination of the active command and other previously scheduled queued commands.

The command is labeled implicitly by the known initiator and target addresses and the LUN. As long as only one command is active from each initiator, the LU can always reconnect to the correct initiator and pointer set with that triple set of information. It is the responsiblity of the initiator to assure that no more than one command is active at any time. Section 6.5.1 **[See proposal X3T9.2/87-51]** describes the actions taken if more than one command is activated from an initiator to the same LUN.

Untagged command queueing can be supported by SCSI-2 devices or by any SCSI X3.131-1986 device meeting conformance level 2.

It is assumed that the initiator will support the required number of active subchannel functions. Each subchannel is effectively a storage area for the pointers associated with each ongoing untagged queued command.

### 6.6.2 Tagged Command Queueing

Tagged command queueing allows a Logical Unit to continue accepting commands until its command queue is full, regardless of how many commands may already be active from each initiator.

Three extended messages, UNORDERED QUEUE TAG, ORDERED QUEUE TAG, and HEAD OF QUEUE TAG are defined to allow the initiator to uniquely label each command or set of linked commands with a distinctive QUEUE TAG logical identifier immediately after the IDENTIFY message is transferred. This allows the initiator to explicitly expand the implicit labeling of the commands so that a reconnection can always be properly identified by the quadruple of information including the initiator address, the target address, the LUN, and the Tag Value provided in the queueing message. Each initiator must assure that all its outstanding

quadruples of labels are unique. The three extended messages each specify a different relationship between the labeled command and other previously enqueued commands.

For a particular initiator, multiple commands labeled by any of the three tagging messages, UNORDERED QUEUE TAG, ORDERED QUEUE TAG, or HEAD OF QUEUE TAG as well as no more than one untagged command may be executed to the same LU. If only commands labeled by UNORDERED QUEUE TAGS are being transmitted, the commands may be executed in an arbitrary order selected by the target device. Commands from other initiators are also executed in an arbitrary order. The command ordering is done by the target to meet the performance and functional goals desired for that target and LU.

Commands with ORDERED QUEUE TAGS must be executed in the exact order received with respect to other ORDERED QUEUE TAG labeled commands and with respect to commands without logical identifiers. In addition, all UNORDERED QUEUE TAG labeled commands, regardless of initiator, received before a particular ORDERED QUEUE TAG labeled command must be executed before that ORDERED QUEUE TAG labeled command. All UNORDERED QUEUE TAG labeled commands, regardless of initiator, received after a particular ORDERED QUEUE TAG labeled command must be executed after that ORDERED QUEUE TAG command. All HEAD OF QUEUE TAG labeled commands are placed first in the queue for execution immediately after any command that may have previously been dequeued and initiated. Consecutive HEAD OF QUEUE TAG labeled commands are executed in FIFO [LIFO?] order. The specification does not define any rules on the relative order of execution of untagged commands and UNORDERED QUEUE TAG labeled commands. Such commands are executed in an order defined by the queue management algorithms in the target.

Commands that do not have tag values provided are managed by the target according to the rules for untagged queueing. Only one such command may be active at a time for each initiator/target/LUN triple. No time relationship is determinable between the execution of untagged commands sent from one initiator and the execution of untagged commands sent from another except through the reservation process.

Links of commands are tagged by a single queue tag. That queue tag applies to the entire chain, but the commands are executed one by one. As each command of the link is terminated, the newly received linked command is placed on the queue using the same type of queue tag and using the same Tag Value. In this manner, commands within a link are executed in strict order, but with no predictable timing relative to commands outside the link, except timings forced by software controlled locking mechanisms.



The RESERVE and RELEASE commands are normally executed as ORDERED QUEUE TAG or HEAD OF QUEUE TAG labeled commands. Caution must be used in programming these commands to avoid constructing queues with deadlocks. An external communications port may be required to prevent such queue contention.

The TEST UNIT READY, INQUIRY, and REQUEST SENSE commands are normally executed as HEAD OF QUEUE TAG labeled commands, since the information they have to offer is often either instantaneously available or has no effect on the state of the LU.

Error information from errors that occur during execution of a command tagged with an initiator/target/LUN/Tag Value quadruple is recovered by execution of a REQUEST SENSE command tagged with the same quadruple. The quadruple is used to uniquely identify the necessary buffered error information. The REQUEST SENSE command is normally labeled by a HEAD OF QUEUE TAG message, since the error information is usually available instantly from the target's error buffers and since the error information is usually required quickly for prompt error recovery.

Devices not supporting any tagging messages, either because they do not support command queueing or because they meet X3.131-1986, respond with a MESSAGE REJECT message to reject the tagging messages. The command is expected to continue from that time on in the normal manner without making use of the tagging information or the Tag Value and without performing tagged queueing. Untagged queueing will still operate normally. Tagged command queueing may also be switched off by the device during certain initialization periods or to control internal resource utilization. If the period of suspension is short, the QUEUE FULL message may be used to temporarily reject commands. If the period is expected to be longer, the MESSAGE REJECT message will allow the commands to continue to be executed but without queueing.

The ABORT TAG message is a mechanism designed to remove selected operations from the queue. The message is presented to the target immediately after the IDENTIFY message to complete the Initiator/Target/LUN/Tag Value quadruple required to completely identify the command to be deleted from the queue. After completing the acceptance of the ABORT TAG message, the target shall go to BUS FREE. Execution of an ABORT TAG message for which there is no corresponding command in process or enqueued in the target is not an error. If the target is able to identify the command referenced by the quadruple of information, then the command will be removed from the queue. If the command is already dequeued and being executed, the command execution is halted. Any pending status or accumulated data is abandoned and not presented to the initiator. The command

may already have started to modify the data in the target. It is the responsibility of the initiator software to perform appropriate system level recovery procedures.

The CLEAR QUEUE message is a mechanism designed to remove all commands labeled by the selecting Initiator/Target/LUN triple from the queue and from execution, regardless of the Tag Value of the command. Any untagged command for that triple is also removed from the queue or from execution. After completing the acceptance of the CLEAR QUEUE message, the target shall go to BUS FREE. Execution of a CLEAR QUEUE message when no commands are in process or enqueued in the target is not an error. Pending status or accumulated data is abandoned and not presented to the initiator. One or more of the commands may already have started to modify the data in the target. It is the responsibility of the initiator software to perform appropriate system level recovery procedures.

### 6.6.3 Example of Command Queueing

An example of command queueing requires the consideration of the execution of a number of commands. After each command, the state of the queue kept in the target must be shown to indicate the function actually performed by the queueing.

#### 6.6.3.1 Typical Tag Sequences for a Queued Command

A tagged queued command requires the following sequences for normal execution. The initiator first arbitrates for the bus, and after successfully obtaining the bus, selects the appropriate target SCSI device. The ATN line is asserted during the selection sequence to indicate that a message out sequence is required by the initator. The first message byte transfered is a normal IDENTIFY message out. The ATN line continues to be asserted during the message out sequence to indicate that the initiator requires a second message out cycle. The second message byte transfered is an EXTENDED MESSAGE message out. The message then requires the transfer of the following length byte, which indicates that two subsequent bytes are to be transferred. The next message out byte contains the code indicating that the EXTENDED MESSAGE is an UNORDERED QUEUE TAG message. The last message out byte is transmitted containing the Tag Value. It is transferred with ATN deasserted to indicate that no more message out bytes are required. The target then obtains the Command Descriptor Block. Assuming the command requires disconnection, the target transmits a DISCONNECT message to the initiator and then enters the BUS FREE phase. The target places the transmitted command, identified by the Initiator/Target/LUN/Tag Value quadruple, at the appropriate place in the queue for execution.

When the target dequeues the command for execution, a physical latency period may be required. At the end of this



period, when the target is prepared to transmit the appropriate data, the target begins an arbitration and enters a RESELECTION phase. After a successful reselection, the target transmits the IDENTIFY message to indicate to the initiator the LU for which the reselection is taking place. After transmitting the IDENTIFY message, the target transmits the four bytes of the EXTENDED MESSAGE, UNORDERED QUEUE TAG, with the Tag Value originally transmitted by the initiator. The initiator uses the Initiator/Target/LUN/Tag Value quadruple to identify the correct set of pointers and control blocks associated with the command and to establish the necessary preconditions for data transfer. The target begins data transfer. With appropriate synchronous or asynchronous data transfer protocols, the required data fields are transmitted to or from the initiator by the target. When the data transfer is successfully completed, the target terminates the operation with GOOD status and with a COMMAND COMPLETE message, indicating that all transfer for the command is completed and that the initiator can post a completion indication for the command to the appropriate host software. The system level completion indication identifies the command through the proper mapping of the Initiator/Target/LUN/Tag Value quadruple to the system level operation identifier.

#### 6.6.3.2 Example of Tagged Queueing Execution

An example of the execution of five queued commands is described in sufficient detail to demonstrate how tagged queueing operates. All tagged commands are from one initiator to a single LU of a single target. The five commands are defined in table 6-8. The target is a direct access device. At the time the commands are first being executed, it is assumed that the actuator is in position to access logical block 10000.

TABLE 6-8
TAGGED QUEUEING EXAMPLE: COMMANDS TO BE EXECUTED

| COMMAND | TAG | TAG VALUE | FIRST BLOCK | BLOCK COUNT |
|---|---|---|---|---|
| READ | UNORDERED | 01 | 10000 | 1000 |
| READ | UNORDERED | 02 | 100 | 1 |
| READ | ORDERED | 03 | 1000 | 1000 |
| READ | UNORDERED | 04 | 10000 | 1 |
| READ | UNORDERED | 05 | 2000 | 1000 |

The optimum order would require that those blocks close to the actuator be the first blocks accessed, followed by those increasingly far from the actuator. However, the command with Tag Value 3 is identified by an ORDERED QUEUE TAG, so

that all unordered tagged commands transferred previously must be executed before, while all unordered tagged commands transferred after the ordered command must be executed after the ordered queue tagged command.

The actual order in which the operations were dequeued and executed would then be as shown in Table 6-9.

TABLE 6-9
TAGGED QUEUEING EXAMPLE: ACTUAL ORDER OF COMMAND EXECUTION

| COMMAND | TAG | TAG VALUE | FIRST BLOCK | BLOCK COUNT |
|---|---|---|---|---|
| READ | UNORDERED | 01 | 10000 | 1000 |
| READ | UNORDERED | 02 | 100 | 1 |
| READ | ORDERED | 03 | 1000 | 1000 |
| READ | UNORDERED | 05 | 2000 | 1000 |
| READ | UNORDERED | 04 | 10000 | 1 |

Commands tagged as 01 and 02 are executed in the order received since the actuator is already in position to execute command 01. Command 02 must be executed before command 04 or 05 because the ordered command tagged as 03 was transmitted after commands 01 and 02 but before commands 04 and 05. Command 03 is then executed after command 02. The commands tagged as 04 and 05 are executed after the ordered command 03. Command 05 is executed before command 04 because the actuator is in position to access block 2000 after executing command 03. Command 04 is then executed.

As an example of the operation of the head of queue command tagging, consider that a new command, identified by a HEAD OF QUEUE TAG message with a Tag Value of 08, is transmitted to the target while the ordered command with the Tag Value of 03 is being executed. The command tagged 03 continues operation, but the new head of queue command is placed in the queue for execution before all subsequent commands. In this case, the queue for execution after the ordered command 03 was executed would appear as shown in Table 6-10.

TABLE 6-10
TAGGED QUEUEING: ORDER MODIFIED BY HEAD OF QUEUE TAG

| COMMAND | TAG | TAG VALUE | FIRST BLOCK | BLOCK COUNT |
|---|---|---|---|---|
| READ | ORDERED | 03 | IN EXECUTION | |
| READ | HEAD OF Q | 08 | 0 | 8 |
| READ | UNORDERED | 05 | 2000 | 1000 |
| READ | UNORDERED | 04 | 10000 | 1 |



The extraction of maximum performance gains using tagged queueing requires careful implementation of the queueing and dequeuing algorithms in the target. In addition, all operating system software in the initiators must be well disciplined to allow a maximum number of ordered commands to be executed with a minimum of synchronizing ordered commands. RESERVE/RELEASE, SET LIMITS, and appropriate software locking conventions must be used to guarantee the proper relationship between transaction execution and the data stored on the peripheral devices. These conventions are not defined by this standard.

The above text should completely define the queueing function for SCSI-2 devices. The committee offers this proposal for your review, modification, and acceptance.