Printer Working Group – 1394 Printing

PWG 1394 Transport Command Set Proposal Revision 0d

Gregory A. Shue, Hewlett-Packard

August 7, 1998

1 Scope

This is an SBP-2 command set addendum proposal for the PWG 1394 Transport protocol. This document is limited to information needed to fully describe the SBP-2 ORB and Status Block data structures, the Commands, Parameters, and policies necessary for providing the required transport functionality. This does not cover Configuration ROM information.

2 Device Model

The device model is defined in the PWG Transport draft document. This paragraph is for context only. The PWG Transport Protocol is an addendum to the SBP-2 protocol, revision 4. Since the PWG Transport needs to simultaneously process data transfers in both directions, the target implementing the PWG Transport must report itself to SBP-2 as using the UNORDERED execution model. This allows the target to complete Task execution however it needs to. Within the PWG Transport target are two ordered execution queues. Each queue maintains an ordered execution sequence of Tasks for a single direction of data flow.

In order to setup the bi-directional communication queues, the initiator must complete each of the following commands in sequence:

a) TRANSPORT_CAPABILITIES – to retrieve the transport-layer capabilities provided by the target. This is necessary for the initiator to find the maximum task set size cacheable by the target.

b) TRANSPORT_OPEN – to configure the transport to use the reduced set of capabilities indicated by the initiator, to cause the target to allocate any necessary resources, and to attempt to establish a connection with the target's transport client.

Once the connection has been established, the initiator may issue the following commands for data to be transferred, provided there is room on the active task list. All transport client data shall be referenced by the ORB's data_descriptor.

- TRANSPORT_I2T_DATA – to request the target to transport the data in the buffer associated with the ORB to the target's client.
- TRANSPORT_T2I_DATA – to request the target to transport, when available, any data available from the target's client to the buffer associated with the ORB.

If the initiator wishes to stop accepting T2I data before disconnecting, it shall issue a TRANSPORT_CLOSE on the queue associated with TRANSPORT_T2I_DATA. This is necessary to allow the target to indicate the transport failure to the target's transport client, so that it may take appropriate action.

If the initiator wishes to stop generating I2T data before disconnecting, it shall issue a TRANSPORT_CLOSE on the queue associated with TRANSPORT_I2T_DATA. This is necessary to allow the target to indicate the End-Of-File to the target's transport client, so that it may take appropriate action.

When the initiator is ready to completely close the connection, it shall explicitly Logout. It may do this without having either direction closed. An initiator may not close both directions and re-open the connection without executing a Logout and Login. This is explicitly to prevent an initiator from monopolizing the target, which is a shared resource on the bus.

This device model does not support ABORT TASK or LOGICAL UNIT RESET management agent functions. Doing so requires unnecessary complexity in the management of the target's internal ordered-execution queues and opens issues with the in-order delivery of data.

TO BE DETERMINED – What should be done about the following case? An initiator opens a connection, carries on a conversation, closes both queues for communication, and then does a TARGET RESET? Presumably, these would allow another TRANSPORT_OPEN to be issued without having the initiator Logout, though it would affect all other initiators logged in to that target by generating a UNIT ATTENTION condition.

3 Data structures

There are three data structures described by this standard:

- SBP-2 Operation request blocks (ORBs);
- SBP-2 Status blocks;
- Command parameter encoding.

3.1 Operation Request Blocks (ORBs)

All ORB formats described in SBP-2, rev 4, shall be supported.

3.1.1 Command Block ORB Format

Command block ORBs are used to encapsulate data transfer or device control commands for transport to the target.

The general format of the command block ORB is illustrated by the following figure.

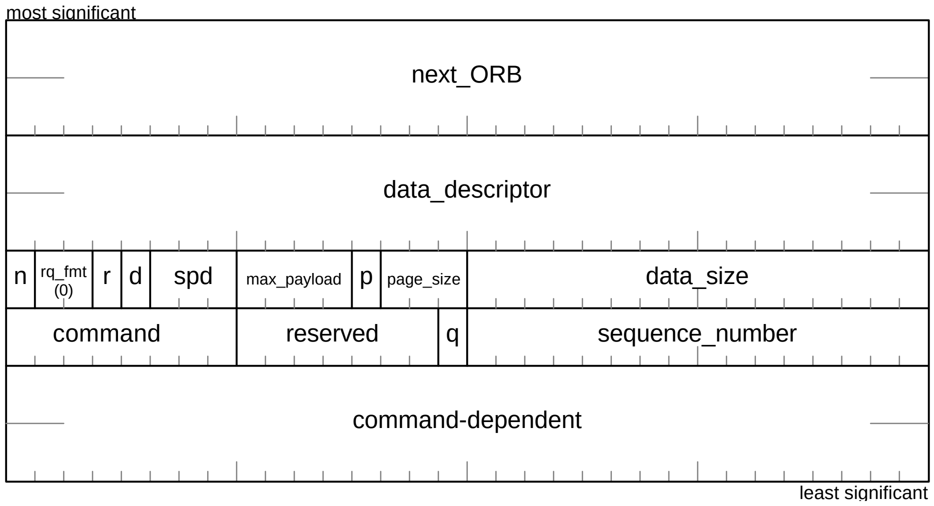

Figure 1 – Command block ORB

The *next_ORB*, *data_descriptor*, *rq_fmt*, *spd*, *max_payload*, *page_size* and *data_size* fields and the *notify*, *direction* and *page_table_present* bits (abbreviated as *n*, *d* and *p*, respectively, in the figure above) shall be as specified by SBP-2.

The *queue_ID* bit (abbreviated as *q* in the figure above) shall identify the ordered execution queue to which the ORB belongs. See clause 8 Queue Management for more information on the target's management of the ordered execution queues.

The *command* field shall specify the command to be executed by the target. The *command* field shall be interpreted according to the table below. All unlisted values are reserved

Value	Command	Queue ID	Description
0	TRANSPORT_CAPABILITIES	1	The target returns default setup information in the buffer
1	TRANSPORT_OPEN	0	The target updates its current parameters according to information fetched from the buffer and opens the connection
2	TRANSPORT_I2T_DATA	0	The target reads data from the buffer
3	TRANSPORT_T2I_DATA	1	The target writes data to the buffer
4	TRANSPORT_CLOSE	either	The target is aware that the initiator will queue no additional ORBs on the associated queue.

The *sequence_number* field carries the initiator-assigned sequence number for this command instance. The sequence numbers are private to each queue. See clause 9 Sequence Number Management for more information.

The *command*-dependent field shall contain information determined by the value of *command*.

3.2 Status block

A target shall store status at an initiator *status_FIFO* address when a request completes and either the notification bit is set, an error occurred, or the completion notification contains other than zero bits beyond the first two quadlets. The *status_FIFO* address is obtained implicitly from the fetch agent context. Whenever the target has status to report, it shall store all or part of the status block shown below.

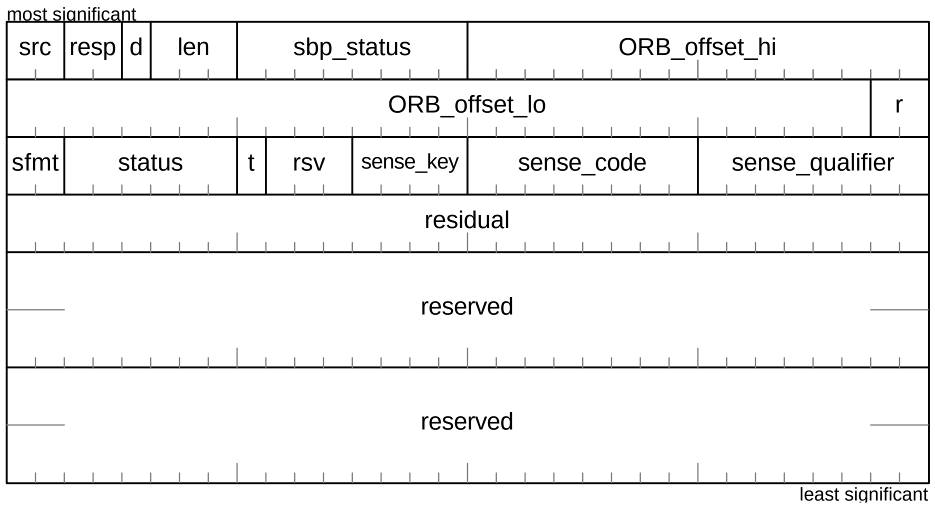

Figure 2 – Status block format

The target shall store a minimum of eight bytes of status information and may store up to the entire 32 bytes defined above so long as the amount of data stored is an integral number of quadlets. A truncated status block shall be interpreted as if the omitted fields had been stored as zeros. The target shall use a single Serial Bus block write transaction to store the status block at the *status_FIFO* address.

The *src*, *resp*, *len*, *sbp_status*, *ORB_offset_hi*, and *ORB_offset_lo* fields and the *dead* bit (abbreviated as *d* in the figure above) shall be as specified by SBP-2.

The *sfmt* field shall specify the format of the status block. The table below defines permissible values for *sfmt*.

Value	Description
0	Current error; status block format defined by this standard
1, 2	Reserved for future standardization
3	Status block format vendor-dependent

The *status* field shall contain status information as defined by this standard. The status field shall only be value provided no SBP-2 errors are reported. The receipt of any status shall indicate that the associated task has ended. The following table defines permissible values for *status*.

***status* value**	**Description**
0	GOOD
2	CHECK CONDITION
$3F_{16}$	Unspecified error
All other values	Reserved for future standardization

Definitions for each status code are given below:

GOOD. This status indicates that the target has successfully completed the task.

CHECK CONDITION. This status indicates that the target has detected a condition that has stopped the Fetch Agent and implicitly aborted the active task set.

UNSPECIFIED ERROR. This status indicates that the target has detected an error condition not specified in this standard. (E.g. an internal error condition which prevents the target from continuing without a power cycle.)

The *tag* bit (abbreviated as *t* in the figure above) shall specify the association of the data returned in the data buffer (out-of-band). This field is reserved for TRANSPORT_I2T_DATA, TRANSPORT_CAPABILITIES, and TRANSPORT_OPEN commands. When the *tag* bit is zero, the contents of the data buffer are part of the normal data stream. When the *tag* bit is one, the contents of the data buffer shall be understood to be distinct from but synchronous with the normal data stream.

The *sense_key, sense_code,* and *sense_qualifier* fields shall contain command completion information defined in this standard.

The contents of the *residual* field are unspecified if the *sfmt* field has a value of three. For *sfmt* values of one or two, the *residual* field is reserved. If the *sfmt* field has a value of zero, this contains the residue of the requested data transfer length minus the length of actual data to be transferred, in bytes, of the command. Negative values are indicated in two’s complement notation.

3.3 Parameter Encoding

All parameters passed in the buffer associated with the ORB (whether to set or return values) shall be encoded using the format that follows. Parameter encodings may be packed together to form a list, which may be passed by reference, when supported by the command.

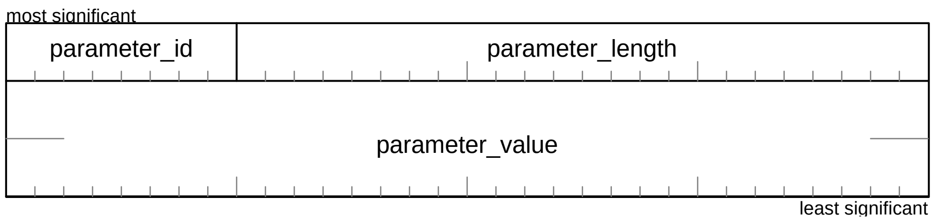

Figure 3 – Parameter ID and value format

The *parameter_id* value shall indicate what type of information is encoded in the *parameter_value* field.

The *parameter_length* field shall indicate the size (in bytes) of valid data in the *parameter_value* field.

The *parameter_value* field contains the actual data for the identified parameter. The data may be of an arbitrary byte length greater than zero. The data shall be padded with reserved bits to the next quadlet boundary. The *parameter_length* field shall not be adjusted to include this padding.

Data shall be stored in the least significant bits of the *parameter_value* field.

4 Standard Parameters

The following table describes all the standard transport parameters. All unspecified values are reserved. The target shall provide all parameters in the TRANSPORT_CAPABILITIES command response. The initiator may restrict the range specified by a parameter for this login by providing a reduced value with the TRANSPORT_OPEN command.

ID	Parameter	Access	Size	Description
1	MAX TASK SET SIZE	RO	14 bits	Returns the maximum number of pending commands across both command queues.
2	MAX I2T DATA SIZE	RW	31 bits	The maximum data size supported for TRANSPORT_I2T_DATA commands.
3	MAX T2I DATA SIZE	RW	31 bits	The maximum data size generated within TRANSPORT_T2I_DATA commands.

4.1 MAX TASK SET SIZE

This parameter returns the maximum size of the active task set across all ordered queues supported by this connection. The initiator must avoid queuing more tasks than this on the Task List. All targets shall support at least one active task for each ordered queue. This value shall be fixed for the duration of a Login.

4.2 MAX I2T DATA SIZE

This parameter indicates the maximum amount of data that may be provided to the target in a TRANSPORT_I2T_DATA ORB. A value of zero (0) for this parameter indicates the target shall not generate data for the initiator.

NOTE – This size is limited by the target's ability to discard data within an active TRANSPORT_I2T_DATA command after being interrupted by a Bus Reset. Data sizes larger than this would prevent the target from discarding all the data referenced by this ORB in the event that this task was aborted either explicitly or implicitly.

NOTE – Because the initiator is required to maintain in-order data flow, the integrity of the data stream must be maintained across command failures and Bus Reset events. For stream services, the target could maintain a fetch offset into the data stream referenced from the last released data. Thus, this parameter is not necessary for stream services.

NOTE – For message services, the parameter is not needed (because both transport clients using the service know the constraints on the message sizes), but the requirement to not retain (or process) any of the data contents must be maintained.

TO BE DETERMINED – Will this parameter be removed?

4.3 MAX T2I DATA SIZE

This parameter indicates the maximum amount of data that will be provided by the target in response to a TRANSPORT_T2I_DATA command. This allows the initiator to adjust its memory usage to better match the target’s capabilities. A value of zero (0) for this parameter indicates the target will not generate data for the initiator. This value shall be fixed for the duration of a Login.

NOTE – This size is limited by the target’s ability to retransmit data within an active TRANSPORT_T2I_DATA command after being interrupted by a Bus Reset.

TO BE DETERMINED – Because this parameter is not required for operation, it needs to be determined whether it shall be removed, left optional, or become required as part of the standard.

5 Error Reporting Precedence

The precedence of error reporting, and the reported values, shall be as follows:

1. SBP-2 errors

2. Unit Attention Condition. If a unit attention condition exists, the target shall report the following condition:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	6 - UNIT ATTENTION	29,0 – POWER ON, RESET, OR BUS DEVICE RESET OCCURRED

3. Data length not supported by protocol (>= 2^{31} bytes). For this error, the target shall report the following condition:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	49.0 – INVALID PACKET SIZE

4. *command* not supported. For this error, the target shall report the following condition:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	26,0 – INVALID COMMAND OPERATION CODE

5. If this command is issued to the wrong queue, then the target shall return status of

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	21.1 – INVALID ELEMENT ADDR

6. Command-specific errors

7. Unknown or unspecified errors

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	0 – NO SENSE	0,0 – NO ADDITIONAL SENSE INFORMATION

6 Commands

6.1 TRANSPORT_CAPABILITIES Command

The initiator shall use this command to request the target to store transport setup information into the buffer. The initiator shall the ORB format shown below.

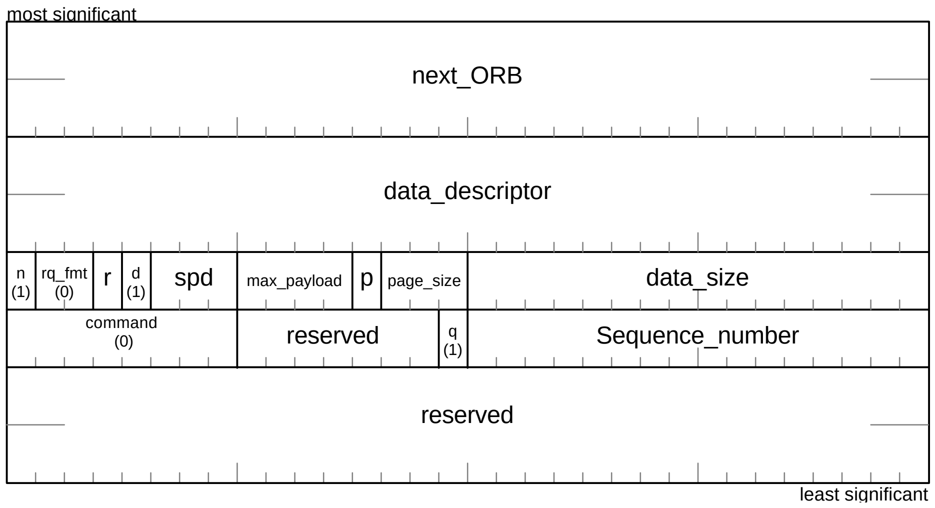

Figure 4 – TRANSPORT_CAPABILITIES Command ORB

All fields shall conform to the descriptions in 3 Command Block ORB Format.

The *notify* bit shall be one.

The *queue_ID* bit shall be one.

The *command* field shall indicate a TRANSPORT_CAPABILITIES command.

6.1.1 TRANSPORT_CAPABILITIES Command Response

The command response shall conform to the format described in 3.2 Status block.

The *sfmt* field shall contain a value of zero (0).

The data associated with this command response shall contain encoding(s), as described in 3.3 Parameter Encoding, for the parameter(s) to be set. All behaviors regarding parameter encoding are described in 3.3 Parameter Encoding

If this command completes successfully, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	0 - GOOD	0 - NO SENSE	0,0 – NO ADDITIONAL SENSE TO REPORT

Else if this command contains an invalid field in the CDB, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	24,0 – INVALID FIELD IN CDB

Else if this command is issued after the connection is open, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	4A,0 – COMMAND PHASE ERROR

Else if this command is issued with a NULL data descriptor, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	21,1 – INVALID ELEMENT ADDRESS

6.2 TRANSPORT_OPEN Command

The initiator shall use this command to request the target to open a connection to the target's transport client using the indicated transport parameters supplied in the buffer. The initiator shall use the ORB format shown below.

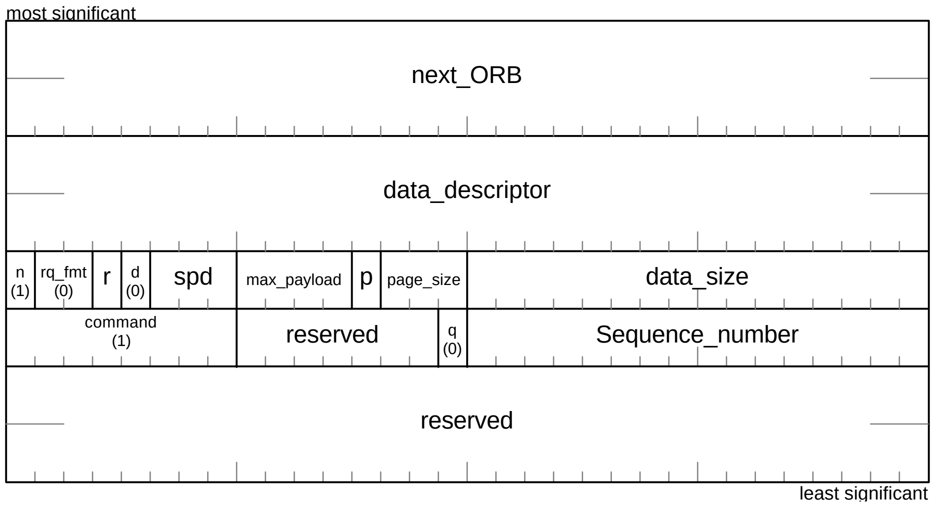

Figure 5 – TRANSPORT_OPEN Command ORB

All fields shall conform to the descriptions in 3 Command Block ORB Format.

The *notify* bit shall be one.

The *queue_ID* bit shall be zero.

The *command* field shall indicate a TRANSPORT_OPEN command.

The data associated with this command shall contain encoding(s), as described in 3.3 Parameter Encoding, for the parameter(s) to be set. All behaviors regarding parameter encodings are described in 3.3 Parameter Encoding.

6.2.1 TRANSPORT_OPEN Command Response

The command response shall conform to the format described in 3.2 Status block.

The *sfmt* field shall contain a value of zero (0).

The *residual* field is reserved for this command.

If this command completes successfully, then the target shall return status of:

$\text{Resp}_{(16)}$	$\text{Sbp\_status}_{(16)}$	$\text{Status}_{(16)}$	$\text{Sense key}_{(16)}$	$\text{ASC}_{(16)}$, $\text{ASQ}_{(16)}$
0 - REQ COMPLETE	0 - NO ADDL INFO	0 - GOOD	0 - NO SENSE	0,0 – NO ADDITIONAL SENSE TO REPORT

Else if this command contains an invalid field in the CDB, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	24,0 – INVALID FIELD IN CDB

Else if this command is issued before a TRANSPORT_CAPABILITIES request has succeeded, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	2C,0 – COMMAND SEQUENCE ERROR

Else if this command contains an invalid parameter encoding, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	1A,0 - PARAMETER LIST LENGTH ERROR

Else if this command contains an unsupported parameter id, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	26,1 – PARAMETER NOT SUPPORTED

Else if this command contains an unsupported parameter value, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	26,2 – PARAMETER VALUE INVALID

Else if this command cannot succeed due to target resource constraints, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	55,0 – SYSTEM RESOURCE FAILURE

Else if this command is issued after the connection is open, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	4A,0 – COMMAND PHASE ERROR

6.3 TRANSPORT_I2T_DATA Command

The initiator shall use this command to request the target to read data from the buffer using the ORB format shown below.

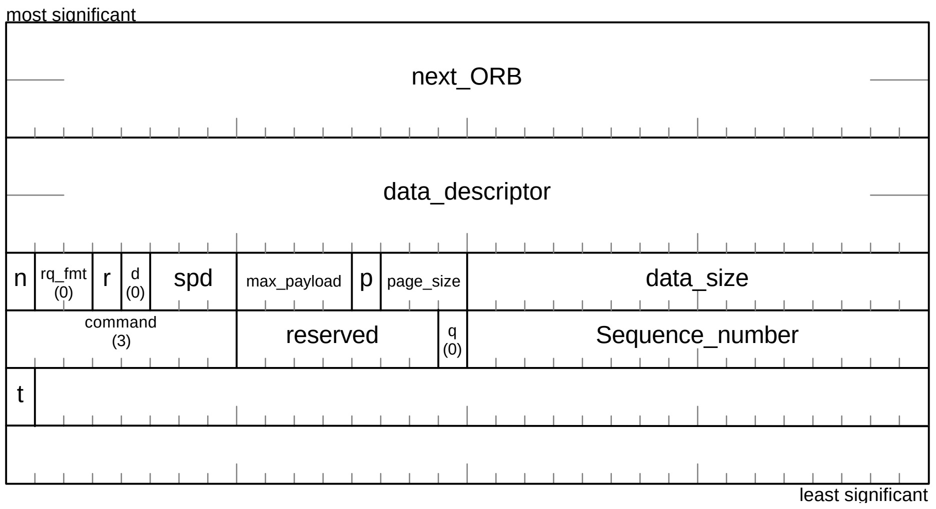

Figure 6 – TRANSPORT_I2T_DATA Command ORB

All fields shall conform to the descriptions in 3 Command Block ORB Format.

Data lengths of zero (0) bytes are valid and shall complete without data being transferred.

The *queue_ID* bit shall be zero.

The *tag* bit (abbreviated *t* in the figure above) shall specify whether or not this data is distinct from the rest of the data stream.

If the command fails, the target shall retain no data referenced by this command.

6.3.1 TRANSPORT_I2T_DATA Command Response

The command response shall conform to the format described in 3.2 Status block.

The *sfmt* field shall contain a value of zero (0).

The *tag* bit in the command response is reserved for this command.

The *residual* field is reserved for this command. Upon successful completion of the command, the target has fetched all the data.

If this command completes successfully, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ

0 - REQ COMPLETE	0 - NO ADDL INFO	0 - GOOD	0 - NO SENSE	0,0 – NO ADDITIONAL SENSE TO REPORT

Else if this command contains an invalid field in the CDB, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	24,0 – INVALID FIELD IN CDB

Else if this command is issued before the connection is open, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	4A,0 – COMMAND PHASE ERROR

Else if the direction is closed (by either the initiator or target), then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	0,2 – END OF PARTITION/MEDIUM DETECTED

6.4 TRANSPORT_T2I_DATA Command

The initiator shall use this command to request the target to TRANSPORT_T2I_DATA to the buffer using the ORB format shown below. The command shall complete when either the requested amount of data becomes available, or the target reaches a transition between tagged and untagged data within the data stream. When the transport is being used for message-style communication, the target shall complete the command when a message boundary is reached. The target may complete the command prior to those events

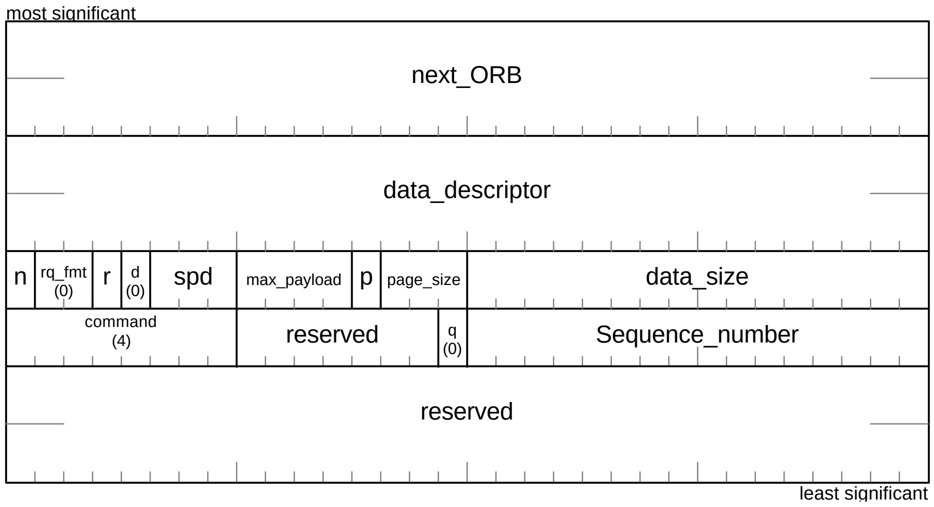

Figure7 – TRANSPORT_T2I_DATA Command ORB

All fields shall conform to the descriptions in 3 Command Block ORB Format.

The *queue_ID* bit shall be zero.

A *data_size* value of zero (0) bytes is valid. In this case, the command shall not complete until some data becomes available for the initiator.

If the command fails the target shall retain all pending data.

6.4.1 TRANSPORT_T2I_DATA Command Response

The command response shall conform to the format described in 3.2 Status block.

The *tag* bit shall specify whether or not this data is distinct from the rest of the data stream. If the data block length is zero, this field is reserved.

For message-based communication, the *residual* field shall indicate either the amount of residual buffer space provided by the command, or the additional amount of data within this message which could not be sent (using twos-complement notation).

For stream-based communication, the *residual* field shall only indicate the amount of unused space in the buffer provided. The target shall not indicate the presence of additional data. Additional data shall be retained for a successive command. A target should complete the command as soon as data is available.

TO BE DETERMINED – When else should a target complete this command? (e.g. “Flush” requested from target’s transport client; the target can no longer retain the data for retransmission?)

If this command completes successfully, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	0 - GOOD	0 - NO SENSE	0,0 – NO ADDITIONAL SENSE TO REPORT

Else if this command contains an invalid field in the CDB, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	24,0 – INVALID FIELD IN CDB

Else if this command is issued before the connection is open, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	4A,0 – COMMAND PHASE ERROR

Else if direction is closed (either by the target or the initiator), then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	0,5 – END-OF-DATA DETECTED

6.5 TRANSPORT_CLOSE Command

The initiator shall use this command to indicate to the target that no more commands will be enqueued on the associated queue. The initiator shall use the ORB format shown below.

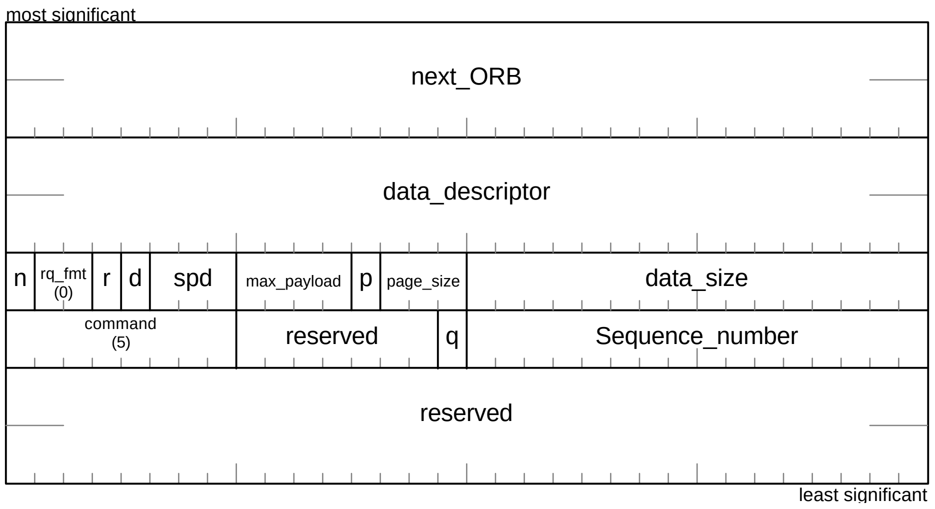

Figure 7 – TRANSPORT_CLOSE Command ORB

All fields shall conform to the descriptions in 3 Command Block ORB Format.

6.5.1 TRANSPORT_CLOSE Command Response

The command response shall conform to the format described in 3.2 Status block.

The *sfmt* field shall contain a value of zero (0).

If this command completes successfully, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
REQ COMPLETE	NO ADDL INFO	GOOD	NO SENSE	0,0 – NO ADDITIONAL SENSE TO REPORT

Else if this command contains an invalid field in the CDB, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
0 - REQ COMPLETE	0 - NO ADDL INFO	2 - CHECK COND	5 - ILLEG REQ	24,0 – INVALID FIELD IN CDB

Else if this command is issued before the connection is open, then the target shall return status of:

Resp	Sbp_status	Status	Sense key	ASC, ASQ
REQ COMPLETE	NO ADDL INFO	CHECK COND	ILLEG REQ	4A,0 – COMMAND PHASE ERROR

7 Status FIFO write request recovery

SPB-2 identifies that a target should take no error recovery actions when it detects a missing acknowledgement after a write to an initiator's status FIFO. It also states that an initiator is expected to discover this error by a higher-level mechanism and to initiate appropriate error recovery actions, though the details are beyond the scope of SPB-2.

When a target detects a missing acknowledgement after a write to an initiator's status FIFO, it does not know which of the following applies:

– The information was correctly received in the status FIFO (and the initiator knows that the command has been completed); or

– The information had a CRC or other error, has not been correctly received by the initiator and the command is still uncompleted.

To recover from this condition, the initiator must determine when to take error recovery actions, and the target must be able to tolerate executed, but uncompleted non-idempotent commands. (Idempotent commands have indistinguishable effect whether they are executed once or many times in succession.)

NOTE – Whatever mechanism is defined needs to be broad enough to cover standard commands, vendor-specific commands, and unsolicited status notifications.

When this condition occurs, the target shall proceed as follows:

a) It shall stop execution of all commands.

b) It shall remember the specific command and queue which had the error.

c) It shall release all resources allocated by the target which are associated with the ORB.

d) It shall update the ORB_POINTER register to reference the offending ORB.

e) It shall transition the Fetch Agent to the DEAD state.

This allows the initiator to detect the erroneous condition by reading the AGENT_STATE register and recover from it by requeuing the uncompleted task list.

Alternatively, the condition would automatically be corrected as a consequence of the next Bus Reset event. Upon completion of a successful reconnection, the initiator must restore TRANSPORT_I2T_DATA and TRANSPORT_T2I_DATA commands and associated data to the task set in the order previously queued within the target. No assumptions shall be made about ORB or buffer locations remaining the same within the 1394 shared memory space, or about the same physical buffers being used.

NOTE – For this device model, this case is assumed to be extremely infrequent, so error recovery actions with a high impact on the bus environment are assumed to be tolerable. It is anticipated to be acceptable for the target's node to generate a Bus Reset when this condition occurs.

8 Queue Management

To simplify queue management and error recovery, the ABORT TASK and LOGICAL UNIT RESET management function shall not be supported.

For stream services, aborted ORBs must be requeued in the same order as previously existed, and must retain the sequence numbers previously assigned.

For message services, requeued ORBs must retain the sequence numbers previously assigned, and must be restored to the same queue order as before.

9 Sequence Number Management

In order for the target to distinguish between completed and acknowledged, executed but uncompleted, and partially or unexecuted commands within a data stream, each command shall contain a queue identifier and associated sequence number. The sequence number shall be unique to the queue.

The sequence number found in the first command placed on each queue shall be used as the initial sequence number value. At any point in time all data block sequence numbers on a queue must lie within one quarter of the size of the sequence number space. Sequence numbers are assigned in ascending order. Once a sequence number has been assigned to a data block, it may not be reused for a different data block until it falls outside the sliding window of usable sequence numbers. Commands, which do not transfer application data still implicitly have a (possibly empty) data block, and so still require a sequence number.

NOTE – For the following discussion Sna represents the sequence number of the last completed and acknowledged command on this queue. Sne represents the sequence number of the last fully executed, but uncompleted (or unacknowledged) command on this queue. Snl represents the sequence number of the command with a lost Ack on a status FIFO write. Sn represents the sequence number of the command being examined. The function dist(A,B) is evaluated as ((A + sequence_number_range – B) % sequence_number_range) and will produces only positive distances.

Whenever a target begins processing a command, it shall examine the sequence number. If the target is trying to recover from a lost status FIFO write Ack:

a) If (dist(Sn, Snl) >= sequence_number_range / 2) then the command had been successfully executed. If the notification bit is set, the command completion notification is immediately given.

b) Else if (dist(Sn, Snl) == 0) then the command has been requeued. The target shall only (re)send the previous completion notification. Upon receipt of the Ack on the status FIFO write, the target shall be synchronized with the initiator.

c) Else if (dist(Sn, Snl) > 0) then the initiator received the previously attempted (successful) completion notification, and the target is resynchronized with the initiator. The sequence number has been implicitly acknowledged.

When the target is not trying to recover from a lost status FIFO write Ack, it shall proceed as follows:

d) If (dist(Sn, Sna) == 0 || dist(Sn, Sna) >= sequence_number_range / 2), then the command has already been executed. If the notification bit is set, the previous completion notification is immediately given.

e) Else the command is executed.

When an initiator queues a command, it may only do so if:

a) The size of the active task set is less than the size of the Maximum active task set supported by the target

b) And, a unused sequence number is available < (sequence_number_range / 2) distance away from the last acknowledged sequence number.

In this case the initiator shall use the lowest available sequence number.

Annex A Change History

Revision 0d (open)

- Renamed the data movement commands to be TRANSPORT_?2?_DATA in order to clarify functionality
- Changed from Set/Get Parameter List to TRANSPORT_CAPABILITIES, TRANSPORT_OPEN, and TRANSPORT_CLOSE to provide open/close delineation for cleaner model-of-use in response to requests from the July 1998 PWG 1394 meeting.
- Removed the *command_oui* and *parameter_oui* fields, and associated vendor-specific references.
- Added in Sequence Numbers for transaction error recovery.
- Rearranged fields in CDB-portion of ORBs for better clarification of sequence number scope.
- Removed Communication Mode parameter, as it is unnecessary with the TRANSPORT_CLOSE command.
- Reorganized the parameter encoding such that *parameter_id* and *parameter_length* are enclosed in a single quadlet.
- Error codes and conditions have been listed for each command.
- The error recovery algorithm proposed by Mr. Shimura (Canon) has been paraphrased and added in.