CONGRUENT SOFTWARE, INC.
98 Colorado Avenue
Berkeley, CA 94707
(510) 527-3926
(510) 527-3856 FAX

FROM: Peter Johansson

TO: T10 SBP-3 working group

DATE: September 20, 2001

RE: Bare-bones isochronous

At the working group meeting in Cupertino in August, 01-222r0 was presented for discussion. That proposal suggested that SBP isochronous operations could be greatly simplified by collapsing the two isochronous fetch agents described in the SBP-3 working draft into a single one responsible for stream operations.

As much as the proposal was received as a step in the right direction, the working group expressed an interest in even more radical simplification. This proposal is an exploration of those ideas. The goals of 01-222r0 remain the same, in that support is envisioned for the following classes of device:

- Simple devices, such as printers or scanners, which neither record (for subsequent playback) nor playback (from prior recording) streams but have a need to use isochronous, instead of asynchronous, mechanisms for data transfer;
- Simple record and playback devices that operate on isochronous streams that contain a single channel; and
- More sophisticated devices that may filter or mix multiple channels during recording or playback.

I think that the addition of the *isochronous* bit to the normal command block ORB and the *plug_index* field to the CREATE STREAM ORB meets the requirements of the first device class. The second device class may be sufficiently supported by these SBP extensions or may require command set-dependent facilities and methods, while the third, most complex, type of device assuredly requires a command set tailored to its needs.

The following normative text changes are proposed for a forthcoming SBP-3 working draft. Sections and clauses to be deleted in their entirety are listed below; the new text (as well as more piecemeal deletions) follows.

Clause	Proposed change
4.8	Replace entire clause with new informative text (yet to be written)
5.1.3	Delete
5.3.3	Delete
5.3.4	Interim request status (new)
9.2a	Stream data transfer (yet to be written)
9.3a	Interim request status (yet to be written)
11	Convert to normative (but optional) annex
12	Delete
G	Delete
H	Change to normative (but optional)

5.1.2.1 Normal command block ORBs

The format of a normal command block ORB with a single data descriptor is illustrated by the figure below.

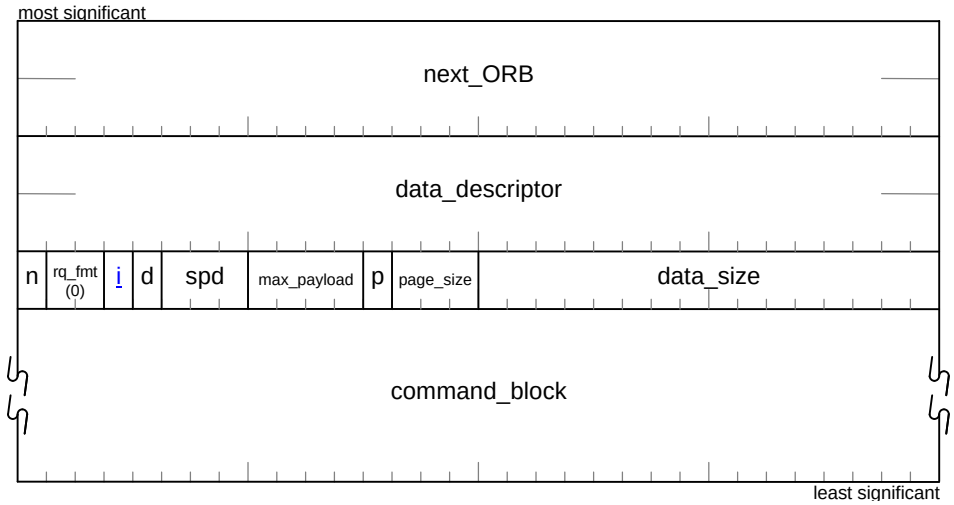

Figure 16 – Normal command block ORB (single data descriptor)

The *next_ORB* field shall contain a null pointer or the address of a dummy ORB or a normal command block ORB and shall conform to the address pointer format illustrated by Figure 12.

The value of the *data_descriptor* field is valid only when *data_size* is nonzero, in which case this field shall contain either the address of the data buffer or the address of a page table that describes the memory segments that make up the data buffer, dependent upon the value of *page_table_present* bit. The format of the *data_descriptor* field, when it directly addresses a data buffer, shall be a 64-bit Serial Bus address or, when it addresses a page table, shall be as specified by Figure 11. When *data_descriptor* specifies the address of a page table, the format of the page table shall conform to that described in 5.2.

The *notify* bit and *rq_fm*t field are as previously defined for all ORB formats. The *rq_fmt* field shall be zero for an ORB which contains a single buffer descriptor.

The *isochronous* bit (abbreviated as *i* in the figure above) specifies the transfer method used for data associated with the ORB. When this bit is zero, Serial Bus read or write transactions shall be used to move data to or from the buffer. Otherwise, when *isochronous* is one, data transfer shall be effected by Serial Bus streams, *i.e.*, packets with a transaction code of A_{16}.

NOTE – The use of Serial Bus streams for the data transfer associated with a command does not preclude the use of *data_descriptor* and its associated fields to describe supplementary or parametric data pertinent to the command. When *isochronous* is one and *data_size* is nonzero, command set-dependent information correlates each data set with its transfer method. For commands that operate on a single data set, the data descriptor fields are ignored when *isochronous* is one.

The *direction* bit (abbreviated as *d* in the figure above) specifies direction of data transfer for the buffer. If the *direction* bit is zero, the target shall use Serial Bus read transactions to fetch data destined for the device medium. Otherwise, when the *direction* bit is one, the target shall use Serial Bus write transactions to store data obtained from the device medium.

The *spd* field specifies the speed that the target shall use for data transfer transactions addressed to the data buffer or page table, as encoded by Table 1.

Table 1 – Data transfer speeds

Value	Speed
0	S100
1	S200
2	S400
3	S800
4	S1600
5	S3200
6 – 7	Reserved for future standardization

The maximum data transfer length is specified as $2^{max\_payload + 2}$ bytes, which is the largest data transfer length that may be requested by the target in a single Serial Bus read or write transaction addressed to the data buffer. The *max_payload* field shall specify a maximum data transfer length less than or equal to the length permissible at the data transfer rate specified by *spd*.

The *page_table_present* bit (abbreviated as *p* in the figure above) shall be zero if *data_descriptor* directly addresses the data buffer. When *data_descriptor* addresses a page table, this bit shall be one.

The *page_size* field shall specify the underlying page size of the data buffer memory. A *page_size* value of zero indicates that the underlying page size is not specified. Otherwise the page size is $2^{page\_size + 8}$ bytes.

When *page_table_present* is one, the *page_size* field also specifies the format of the data structure that describes the data buffer. A *page_size* value of zero implies the unrestricted page table format (also known as a scatter/gather list). Otherwise, a nonzero *page_size* indicates a normalized page table.

If *page_table_present* is zero, the *data_size* field shall contain the size, in bytes, of the system memory addressed by the *data_descriptor* field. Otherwise *data_size* shall contain the number of elements in the page table addressed by *data_descriptor*.

The *command_block* field contains information not specified by this standard.

NOTE – The normal command block ORB with dual data descriptors is omitted from this proposal because it is unaltered from SBP-3 Revision 1d.

5.1.4.3 Create stream ORB

Before any stream requests are made of a target, the initiator shall first complete a create stream procedure that uses the ORB format shown below.

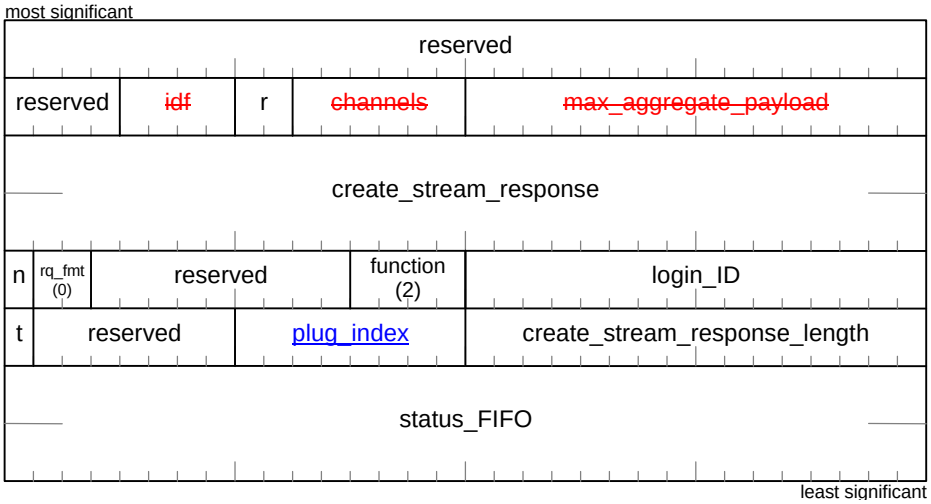

Figure 27 – Create stream ORB

The *idf* field specifies the format of recorded isochronous data. Valid values for *idf* are shown in the table below.

Value	Data format	Description	Reference
0	Unspecified	The data format is determined by the command set or device type and is beyond the scope of this standard	
1	—	Reserved for future standardization	
2	Isochronous data interchange (without cycle mark indices)	The data is structured by cycle marks, which are recorded on the medium.	Section 11 (also 12.2.3)
3	Isochronous data interchange (with cycle mark indices)	The data is structured by cycle marks which are recorded on the medium; a cycle mark index is present every 512 bytes.	Section 11 (also 11.3 and 12.2.3)
4—F_{16}	—	Reserved for future standardization	

The *channels* field specifies the maximum number of isochronous channels that are to be simultaneously transmitted or received.

The *max_aggregate_payload* field is the aggregate maximum isochronous payload that the target is requested to support for the stream. That is, the sum of all the *data_length* fields of Serial Bus isochronous packets transmitted or received for all of the stream's isochronous channels shall not exceed *max_aggregate_payload* in a single isochronous period. The target is not required to enforce this limit.

The *create_stream_response* and *create_stream_response_length* fields specify the address and size of a buffer allocated for the return of the create stream response. The *create_stream_response* field shall conform to the format for address pointers specified by Figure 11. The buffer shall be in the same node as the initiator and shall be accessible to a Serial Bus block write transaction with a data transfer length less than or equal to *create_stream_response_length*. The initiator shall set *create_stream_response_length* to a value of at least ~~24~~12; the target may ignore this field.

The *notify* bit and the *rq_fmt* field are as previously defined for management ORB formats.

The *login_ID* field shall contain a login ID value obtained as the result of a successful login.

The *talker* bit (abbreviated as *t* in the figure above) shall specify the type of isochronous stream requested. If the target resources are to be configured for listening, *talker* shall be zero.

The *plug_index* field may uniquely identify a single source or sink or stream data within the context of the target. If *plug_index* is zero, command set-dependent methods specify the parameters of all stream data. When *plug_index* is in the range one to $1F_{16}$, inclusive, it identifies a plug control register that mediates reception or transmission of stream data. The type of plug control register referenced, either an input or output plug control register, is determined by the value of the *talker* bit. Consult IEC 61883-1 for the specification of plug control registers. Other values of *plug_index* are reserved for future standardization.

The *status_FIFO* field is as previously defined for management ORB formats and shall contain an address allocated for the return of status for the CREATE STREAM request, status for all subsequent requests signaled to either the *stream_command_block_agent* or *stream_control_agent* allocated for this login and any unsolicited isochronous error report(s) generated by the logical unit for this stream.

If the create stream request fails the contents of the response buffer are unspecified. Otherwise, upon successful completion of a create stream request, the response is returned in the format illustrated below.

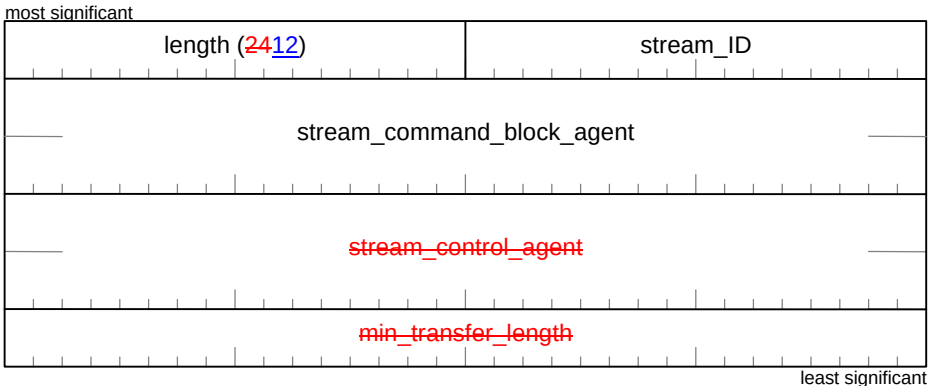

Figure 28 – Create stream response

The *length* field shall contain the length, in bytes, of the create stream response data and shall be equal to ~~24~~12.

The *stream_ID* identifies an isochronous stream for which target resources have been allocated. The initiator shall use this value to identify all subsequent requests directed to the target's management agent that pertain to this stream.

~~The *stream_command_block_agent* and *stream_control_agent* fields specify the base address of the agent's CSRs, which are defined in 6.4. Both fields shall conform to the format for address pointers specified by Figure 11. The *node_ID* portion of either field shall have a value equal to the most significant~~

16 bits of the target's NODE_IDS register. If the target does not implement a stream control agent, the contents of *stream_control_agent* are unspecified and shall be ignored by the initiator.

The *min_transfer_length* field advises the initiator of the minimum *stream_length* value desired by the target in stream command block ORBs in order to sustain the isochronous data transfer rate requested by the login. If the initiator presents any stream command block ORBs whose *stream_length* value is less than this minimum, the target may experience underflow or overflow in isochronous data while talking or listening at the requested rate. Even if this minimum is respected it is still possible for underflow or overflow to occur.

5.3 Status block

A target may store status at an initiator *status_FIFO* address when a request completes (successfully or in error) or because of an unsolicited event (device status change ~~or isochronous error report~~). The *status_FIFO* address is obtained either explicitly from the ORB to which the status pertains or implicitly from the fetch agent context. Whenever the target has status to report and is enabled to do so, it shall store all or part of the status block shown below.

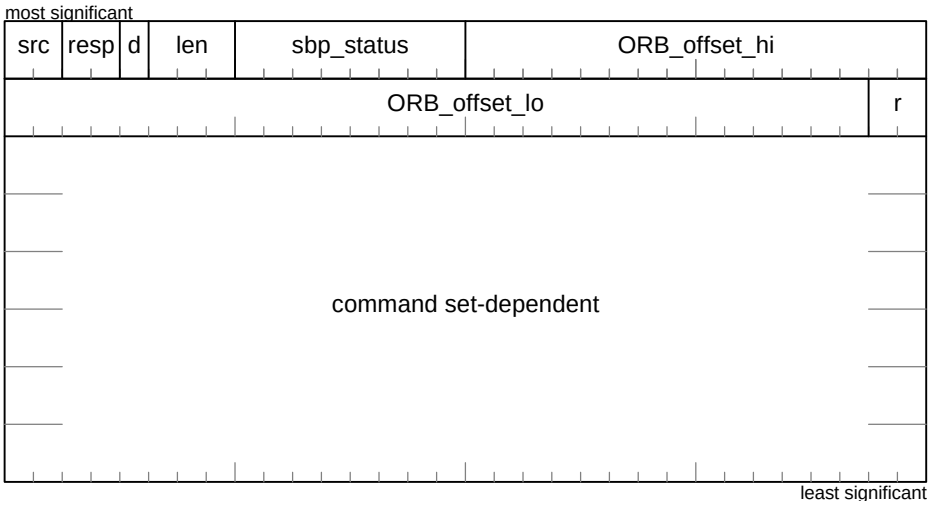

Figure 37 – Status block format

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 and shall store a status block no more than once for the corresponding ORB.

The *src* field indicates the origin of the status block, as specified by the table below.

Value	Description
0	The status block pertains to the ORB identified by *ORB_offset_hi* and *ORB_offset_lo*; at the time the ORB was most recently fetched by the target the *next_ORB* field did not contain a null pointer.
1	The status block pertains to the ORB identified by *ORB_offset_hi* and *ORB_offset_lo*; either the *next_ORB* field is absent or at the time the ORB was most recently fetched by the target the *next_ORB* field was null.
2	The status block is unsolicited and contains device status information; the contents of the *ORB_offset_hi* and *ORB_offset_lo* fields shall be ignored.
3	The status block contains interim request status that pertains to the ORB identified by *ORB_offset_hi* and *ORB_offset_lo* ~~is unsolicited and contains isochronous error report information; the contents of the *ORB_offset_hi* and *ORB_offset_lo* fields are redefined to contain the information~~. The value of the *next_ORB* field at the time the ORB was most recently fetched by the target is unspecified.

The *resp* field shall contain a response status defined in the table below.

Value	Name	Description
0	REQUEST COMPLETE	The request completed without transport protocol error (Either *sbp_status* or command set-dependent status information may indicate the success or failure of the request)
1	TRANSPORT FAILURE	The target detected a nonrecoverable transport failure that prevented the completion of the request
2	ILLEGAL REQUEST	There is an unsupported field or bit value within the first 20 bytes of the ORB
3	VENDOR DEPENDENT	The meaning of *sbp_status* shall be specified by the vendor

The *dead* bit (abbreviated as *d* in the figure above) shall indicate whether or not the target fetch agent transitioned to the dead state upon storing the status block. When *dead* is zero, the reported status has not affected the state of the fetch agent. If the *dead* bit is set to one, the fetch agent transitioned to the dead state as a consequence of the error condition reported by the status block.

The *len* field shall specify the quantity of valid status block information stored at the *status_FIFO* address. The minimum value of *len* is one. The size of the status block is encoded as *len* + 1 quadlets.

The *sbp_status* field provides additional information that qualifies the response status in *resp*. The meanings assigned to *sbp_*status vary according to the value of *src* and *resp* and are described below.

When *src* is zero, one or three, the *ORB_offset_hi* and *ORB_offset_lo* fields together uniquely identify the ORB to which the status block pertains. Otherwise, if *src* is two, the *ORB_offset_hi* and *ORB_offset_lo* fields are ~~either~~ ignored ~~or redefined~~.

For all status block formats, the remainder of the status block after the first two quadlets, up to an overall maximum of 32 bytes, is command set-dependent.

5.3.1 Request status

Upon completion of a request, if the *notify* bit in the ORB is one or if there is exception status to report, the target shall store all or part of the status block shown in Figure 37. For management ORBs (which explicitly provide the *status_FIFO* address as part of the ORB), the target shall store the status block at the address specified. Otherwise (for normal command block ORBs) the target shall store the status block at the *status_FIFO* determined by the fetch agent to which the ORB was signaled. In the case of command block ORBs the initiator provides the *status_FIFO* address as part of the login request while for stream command block and stream control ORBs it is provided in the create stream request.

When *resp* is equal to zero, REQUEST COMPLETE, the possible values for *sbp_status* are specified by the table below. Any value not enumerated is reserved for future standardization.

Value	Description
0	No additional information to report
1	Request type not supported
2	Speed not supported
3	Page size not supported
4	Access denied
5	Logical unit not supported
6	Maximum payload too small
7	Reserved for future standardization
8	Resources unavailable
9	Function rejected
10	Login ID not recognized
11	Dummy ORB completed
12	Request aborted
FF_{16}	Unspecified error

If a Serial Bus error occurs in the transport (*resp* is equal to one, TRANSPORT FAILURE), the *sbp_status* field either shall have a value of FF_{16}, unspecified error, or else the field shall be redefined as illustrated below. This format provides for the return of additional information about the transport failure.

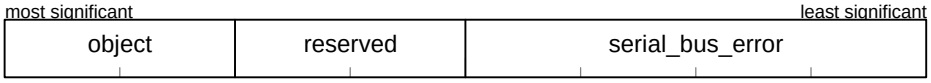

Figure 38 – TRANSPORT FAILURE format for *sbp_status*

The *object* field shall specify which component of an SBP-3 request, the ORB, the data buffer or the page table, was referenced by the target when the error occurred. The value of *object* shall be as defined by the following table.

Value	Referenced object
0	Operation request block (ORB)
1	Data buffer
2	Page table
3	Unable to specify

The *serial_bus_error* field shall contain the error response for the failed request, as encoded by the table below.

Value	Serial Bus error	Comment
0	Missing acknowledge	
1	Reserved; not to be used	
2	Time-out error	An *ack_pending* was received for the request but no response subaction was completed within the time-out limit
3	Reserved; not to be used	
4 – 6	Busy retry limit exceeded	The value reflects the last acknowledge, *ack_busy_X*, *ack_busy_A* or *ack_busy_B*
$7 – A_{16}$	Reserved for future standardization	
B_{16}	Tardy retry limit exceeded	An *ack_tardy* was received for the request and the vendor-dependent retry limit (which may be based upon either time or number of occurrences) for tardy responses has been exceeded
C_{16}	Conflict error	A resource conflict was detected by the addressed node
D_{16}	Data error	The data field failed the CRC check or the observed length of the payload did not match the data_length field
E_{16}	Type error	A field in the request was set to an unsupported value or an invalid transaction was attempted (*e.g.*, a write to a read-only address)
F_{16}	Address error	The destination_offset field specified an inaccessible address in the addressed node

In the cases of conflict error and data error, these are errors that the target may retry up to an implementation-dependent limit before reporting TRANSPORT FAILURE.

No additional information is provided in *sbp_status* when *resp* equals two, ILLEGAL REQUEST. In this case, *sbp_status* shall be set to FF_{16}. An SBP-3 response code of ILLEGAL REQUEST shall not be used to indicate unsupported fields or bit values in the command set-dependent portion of the ORB. This response code shall be used only to indicate an error in the first 20 bytes of the ORB.

The *ORB_offset_hi* and *ORB_offset_lo* fields together form an *ORB_offset* field that uniquely identifies the ORB to which the status block pertains. The target shall form *ORB_offset* from the least significant 48 bits of the Serial Bus address used to fetch the ORB; the least significant two bits shall be discarded.

5.3.2 Unsolicited device status

When a change in device status occurs that affects a logical unit, the target may store the status block shown in Figure 37 at the *status_FIFO* address provided by the initiator as part of a login request (see 5.1.4.1). If a target stores unsolicited status for any initiator logged-in to a logical unit it shall attempt to store status for all initiators logged-in to the same logical unit.

The *src* field shall be one to indicate unsolicited device status.

The *resp* field shall have a value of REQUEST COMPLETE or VENDOR DEPENDENT.

The *dead* bit and the *len* field are as previously defined for the status block.

If *resp* is equal to REQUEST COMPLETE, *sbp_status* shall be zero. Otherwise the content and meaning of *sbp_status* shall be specified by the vendor.

The contents of the *ORB_offset_hi* and *ORB_offset_lo* fields are unspecified and shall be ignored by the initiator.

5.3.3Unsolicited isochronous error report

Upon detection of an isochronous error while talking or listening, if error reporting has been enabled the target shall store the status block shown below at the *status_FIFO* address provided by the initiator as part of the create stream request (see 5.1.4.3).

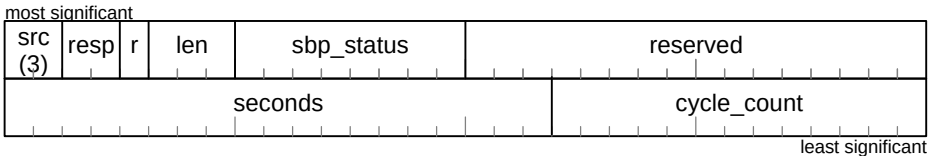

Figure 39 – Unsolicited status format for isochronous errors

The *src* field shall be three to indicate unsolicited status that describes an isochronous error.

The *resp* field shall contain a response status of TRANSPORT FAILURE.

The *len* field shall be equal to one.

The *sbp_status* field shall specify the nature of the isochronous error, as encoded by the table below.

Value	Isochronous error description
0	Reserved (not available for future standardization)
1	Missing CYCLE START packet
2	Data CRC error in received isochronous packet
3	Data length error in received isochronous packet
4	Internal underflow with the result that recorded isochronous data was not transmitted on Serial Bus
5	Internal overflow with the result that isochronous data observed on Serial Bus was not recorded on the medium
6	Format error in recorded isochronous data with the result that data was not transmitted on Serial Bus
7	Data payload in recorded isochronous data too large for transmission on Serial Bus at the requested speed
8 — FE_{16}	Reserved for future standardization
FF_{16}	Unspecified error

The *seconds* field shall contain the least significant 19 bits of the BUS_TIME register at the time of the isochronous stream error.

The *cycle_count* field shall contain the cycle count, between zero and 7999, at the time of the error. The cycle count shall be obtained from the target's free-running cycle timer and shall not be latched from the last observed CYCLE START packet.

5.3.4 Interim request status

Prior to the completion of a request, the target may store all or part of the status block shown in Figure 37. For management ORBs (which explicitly provide the *status_FIFO* address as part of the ORB), the target shall store the status block at the address specified. Otherwise (for normal command block ORBs) the

target shall store the status block at the *status_FIFO* determined by the fetch agent to which the ORB was signaled; the initiator provides the *status_FIFO* address as part of the login or create stream request.

The *src* field shall be three to indicate interim request status.

The *resp* field shall have a value of REQUEST COMPLETE or VENDOR DEPENDENT.

The *dead* bit and the *len* field are as previously defined for the status block.

If *resp* is equal to REQUEST COMPLETE, *sbp_status* shall be zero. Otherwise the content and meaning of *sbp_status* shall be specified by the vendor.

The *ORB_offset_hi* and *ORB_offset_lo* fields together form an *ORB_offset* field that uniquely identifies the ORB to which the status block pertains. The target shall form *ORB_offset* from the least significant 48 bits of the Serial Bus address used to fetch the ORB; the least significant two bits shall be discarded.

Annex H
(~~informative~~ normative)

AV/C Encapsulation

Devices that use the AV/C Digital Interface Command Set use IEC 61883-1 (1998-02) Function Control Protocol (FCP) for the encapsulation of command and status. This annex ~~illustrates~~ specifies how SBP-3 ~~could~~ may be utilized as the transport layer in place of FCP.

AV/C devices are consumer electronic devices such as camcorders, VCRs, stereo tuners and televisions—this is not an exhaustive list. Their commands, status and operations are standardized by the 1394 Trade Association Specification for AV/C Digital Interface Command Set General Specification, Version ~~2.0~~ 4.0, ~~April 22, 1997~~ July 23, 2001. These devices present or accept most of their data over Serial Bus isochronous channels; asynchronous requests are used for control information.

~~AV/C devices differ from the isochronous model presented in 4.7 in two important ways: *a)* there is a single command queue and *b)* isochronous stream control is accomplished by plug control registers (PCRs) and commands contained within the single queue. These differences are illustrative, not only for AV/C devices but also for other, future devices which may use a similar model.~~

~~This annex is particular but the intent is general: by way of an AV/C example, it provides guidance for the use of SBP-3 for any device that follows an equivalent, single-queue model.~~

H.1 Logical unit, unit and subunit models

SBP-3 describes a hierarchical device structure composed of units with subordinate logical units. AV/C has a hierarchical structure of units with subordinate subunits.

An AV/C unit is described by configuration ROM entries in a unit directory and is implemented as a single logical unit. AV/C subunits are not visible as SBP-3 objects; except for queues that result from create stream operations, a single queue is maintained for commands for all the subunits of an AV/C unit.

H.2 AV/C command ~~frame~~ sequence

An AV/C command sequence consists of a command frame delivered to the AV/C device and one or more response frames returned to the controller. Both command and response frames are variable-length data structures between four and 512 bytes in length, and may be described by a normal command block ORB with dual data descriptors ~~A normal command block ORB can accommodate 12 bytes of command frame~~, as illustrated below.

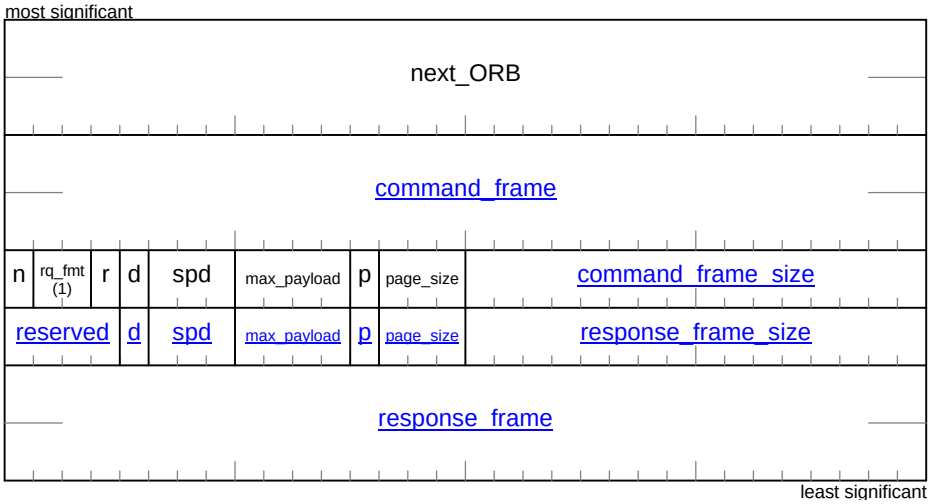

Figure H-1 – AV/C command frame ORB

The *next_ORB*, ~~*data_descriptor,*~~ *rq_fmt*, *spd*, *max_payload*, and *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) are as specified in 5.1.2.1. ~~The AV/C commands described in the Version 2.0 1394 Trade Association specification do not reference a data buffer; consequently the *data_size* field is always zero in an AV/C command frame ORB.~~

The first data descriptor in the ORB, *command_frame*, references a buffer that contains an AV/C command frame in the format specified by IEC 61883-1. The *direction* bit associated with this data descriptor shall be zero. Page tables are not used for AV/C command frames; the *page_table_present* bit shall be zero and *command_frame_size* shall be less than or equal to 512.

The second data descriptor in the ORB, *response_frame*, references a buffer into which the AV/C device may store a response frame in the format specified by IEC 61883-1. The *direction* bit associated with this data descriptor shall be one. Page tables are not used for AV/C response frames; the *page_table_present* bit shall be zero and *response_frame_size* shall be less than or equal to 512.

Because the entire AV/C command sequence consists of the command and response frames, there is no command-dependenet information in the ORB.

~~The *ctype*, *subunit_type*, *subunit*, *opcode* and *operand* fields are specified by AV/C.~~

~~H.3 AV/C response frame~~

~~An AV/C response frame is a variable-length data structure between four and 512 bytes in length. The status block could hold up to 24 bytes of response frame, but since this exceeds the capacity of the AV/C command ORB the size of the status block is restricted to 20 bytes, as illustrated below.~~

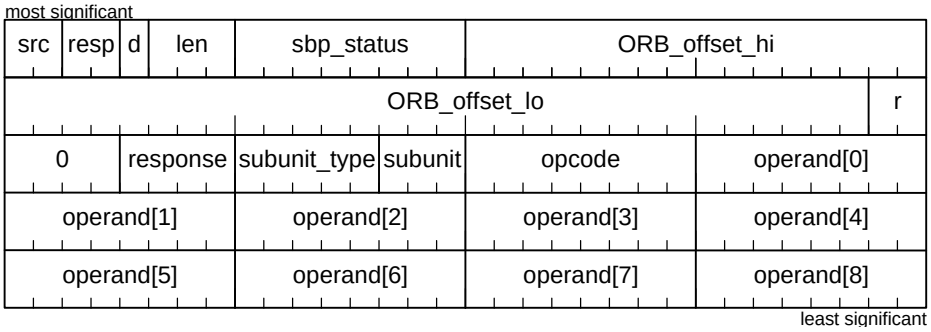

Figure A-2 – Status block format for AV/C response frame

The *src*, *resp*, *sbp_status* fields and the *dead* bit (abbreviated as *d* in the figure above) are as specified in 5.2.3.

The *len* field indicates the size of the status block and has a value of four.

The *ORB_offset_hi* and *ORB_offset_lo* fields together identify the AV/C command frame to which the AV/C response frame pertains. This provides positive command and response matching—an enhanced capability in comparison to FCP.

The *response*, *subunit_type*, *subunit*, *opcode* and *operand* fields are specified by AV/C.

H.4 Configuration ROM

The sample bus information block and root directory for basic targets illustrated in Annex D are equally applicable to targets that use the AV/C command set and are not repeated below. Figure H-3 shows an example of a unit directory and textual descriptor leaves for an AV/C device.

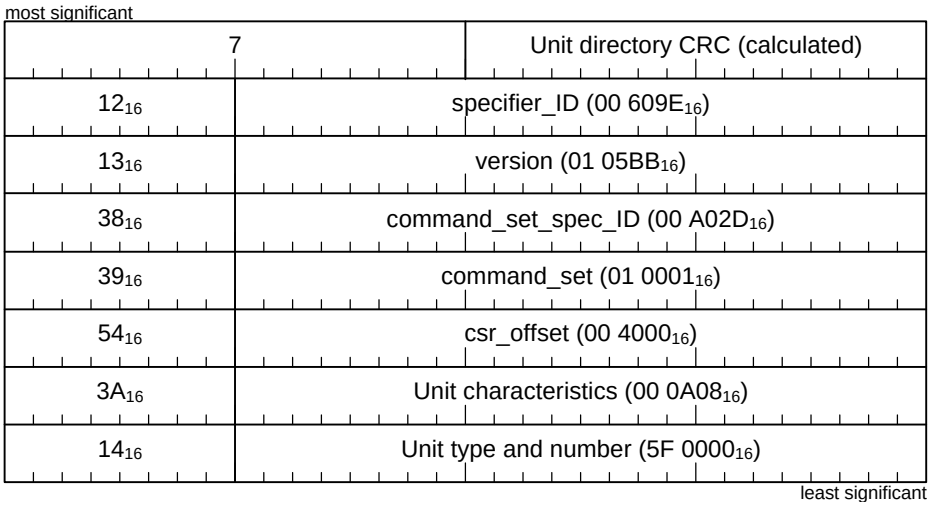

Figure H-3 – AV/C unit directory

The Command_Set_Spec_ID and Command_Set_Version entries, with a *key* field of 38_{16} and 39_{16}, respectively, specify that the target uses the AV/C command set defined by the Version 2.0 1394 Trade Association Specification.

The Management_Agent entry in the unit directory, with a *key* field of 54_{16}, has a *csr_offset* value of 00 4000_{16} that indicates that the MANAGEMENT_AGENT register has a base address of FFFF F001 0000_{16} within the node's memory space.

The Unit_Characteristics entry in the unit directory, with a *key* field of $3A_{16}$, has an immediate value of 00 $0A08_{16}$. This indicates a target is expected to complete a login within five seconds and fetches 32-byte ORBs.

The Logical_Unit_Number entry in the unit directory, with a *key* field of 14_{16}, has an immediate value of 5F 0000_{16}; this identifies logical unit zero and indicates that AV/C commands are used to query the number and type of subunits associated with the unit. It also indicates a target that implements the basic task management model, executes commands in the order queued and does not support ~~(the dual-queue model of)~~ isochronous operations.

H.5 Operations

Control of an AV/C device implemented with SBP-3 follows the steps described in sections 8 and 9. Subsequent to a successful login, the controller may create a queue of one or more ORBs that ~~hold~~ describe AV/C command frames and signal these to the target. As the commands complete, the response frames are returned in the buffers provided by the controller ~~status blocks addressed to the *status_FIFO* specified by the controller at login~~.

~~SBP-3 offers a number of advantages over FCP for the delivery of AV/C command and response frames:~~

- ~~The target may pace command processing to suit its own requirements and resource availability. Consequently the controller should not expect busy responses and does not have to provide error recovery for busy conditions;~~
- ~~The correlation between a particular AV/C command frame and its response frame is explicit and unambiguous—even in the case of complex AV/C devices that may be able to queue or process more than one command at a time;~~
- ~~SBP-3 permits the controller to manage the commands in progress at the target and be certain of the completion status of pending commands if one or more commands are terminated prematurely; and~~
- ~~The AV/C command frame ORB has descriptor fields for a data buffer that permit the target to use a "pull" model for asynchronous data transfer to or from the controller. The target paces data delivery between itself and the controller; FCP has no such model.~~

~~As AV/C command sets are developed for newer and more sophisticated devices, these SBP-3 features may become increasingly important to systems implementers.~~

TO BE DETERMINED – The method used for interim AV/C responses requires working group discussion.