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: August 20, 2001

RE: Stream ORBs: the fewer, the better

For several working group meetings there has been convergent discussion on the desirability of radically simplifying isochronous operations by collapsing the two (presently described) isochronous fetch agents into a single one responsible for stream operations. The text that follows is my attempt to follow this thread to its logical conclusion. I intended to satisfy three usage models:

- 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, not 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 revised stream command block ORB meets the requirements of the first two device classes and the addition of management ORBs for stream control satisfies the third class.

The following normative text is proposed for inclusion in a forthcoming SBP-3 working draft.

5.1.2.2 Stream command block ORB

A stream command block ORB is a structure that has the format illustrated below.

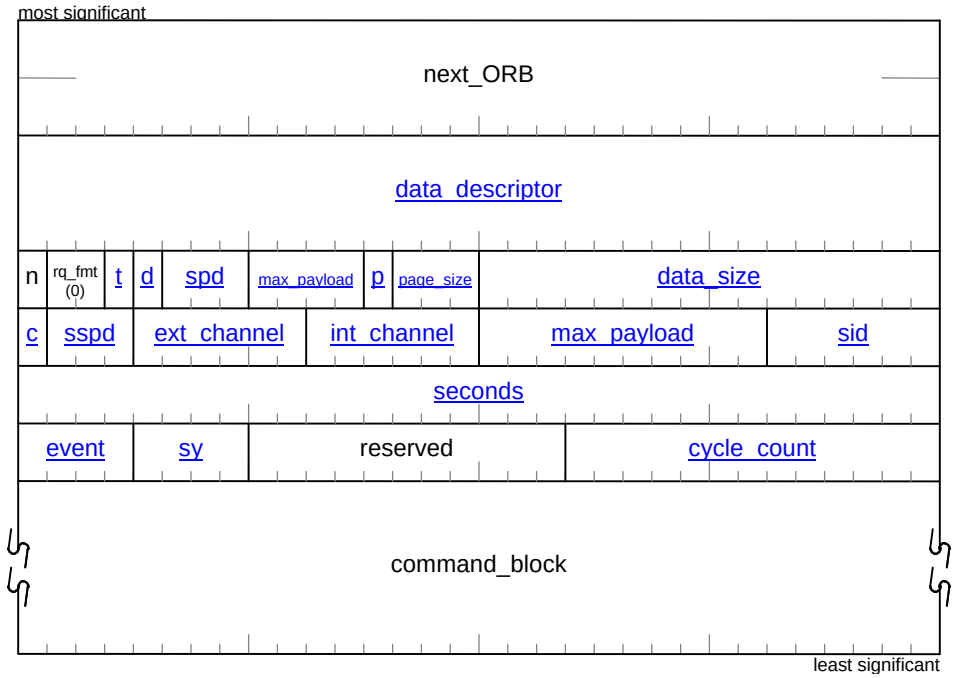

Figure 18 – Stream command block ORB

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

~~The *cm* field (together with the *cycle_mark_offset* field) specifies the location of the first quadlet of isochronous data (stream offset) as encoded by the table below.~~

~~Value~~	~~*cycle_mark_offset*~~	~~Stream offset~~
~~0~~	~~Undefined~~	~~Zero~~
~~1~~	~~Undefined~~	~~*cycle_mark_offset*~~
~~2~~	~~Location of first CYCLE MARK~~	~~Zero~~
~~3~~	~~Location of first CYCLE MARK~~	~~*cycle_mark_offset*~~

~~The stream offset derived from the combination of *cm* and *cycle_mark_offset* specifies the location of the first quadlet of the isochronous data as an offset, in quadlets, relative to the starting medium location indicated by the *command_block*. For a block device, the stream offset, expressed in bytes, shall be less than the block size of the device.~~

~~NOTE – The command transported by the stream command block ORB specifies a starting location on the medium and an associated transfer length. Particularly in the case of block devices, the relevant isochronous data may be a subset of the data length and may commence at a nonzero offset relative to the natural block boundaries of the medium – hence the necessity for the additional values, *stream_length* and stream offset, to completely characterize the request.~~

~~The *cycle_mark_offset* field, when *cm* has a value of two or three, specifies the location of the first CYCLE MARK packet as an offset, in quadlets, relative to the starting medium location indicated by the *command_block*. When *cm* has a value of one, *cycle_mark_offset* specifies the stream offset instead. In either case, the value of *cycle_mark_offset*, converted to bytes, shall be less than *stream_length*.~~

~~NOTE The *cycle_mark_offset* field may be useful to reestablish synchronization within the recorded isochronous data if a prior stream command block terminated in error.~~

~~The *stream_length* field specifies the length of data, in bytes, that is to be transferred to or from the device medium.~~

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 a data buffer or the address of a page table that describes the memory segments that make up a 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.

The *talker* bit (abbreviated as *t* in the figure above) specifies the direction of isochronous data transfer for the command contained within the ORB. When the *talker* bit is zero, data is received from Serial Bus but when the bit is one data is transmitted to Serial Bus.

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

The *spd* field shall specify the maximum speed that the target may use for data transfer transactions addressed to the segment descriptor array, as encoded by Table 1.

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 *sspd* field is valid only if the *talker* bit is one. In this case *sspd* determines the speed at which the isochronous packets shall be transmitted, as encoded by Table 1.

The *ext_channel* field shall specify the channel number observable in the stream's isochronous packets on Serial Bus. When *talker* is zero, the *ext_channel* field provides a filter for packet reception. If *talker* is one, the field specifies the channel number transmitted in isochronous packets.

The *int_channel* field shall specify the channel number used internally by the device for the stream's isochronous packets. Some devices (or some of their operating modes) do not use internal channel numbers, in which case the field is ignored. For other devices, the combination of *int_channel* and *ext_channel* specifies a transformation between internal and external (Serial Bus) channel numbers, dependent on whether the device is listening or talking. When the device is a listener, the observed channel number in the isochronous packet header shall be replaced with *int_channel* as the data is recorded on the medium. When the device is a talker, the channel number obtained from the medium shall be replaced with *dest_channel* at the time the isochronous packet is transmitted.

The *max_payload* field shall specify the largest data payload, in quadlets, the device may transmit in a single isochronous packet whose *channel* field is equal to *ext_channel*. This field is meaningful only when *talker* is one.

The *sid* field is valid only if *target* is one. If the recorded isochronous data conforms to the CIP format described in Annex G, the *sid* field in the previously recorded CIP header shall be replaced with the *sid* value specified when the packet is transmitted—unless the value of the *sid* field is $3F_{16}$, in which case no output transformation shall be made.

The *seconds* and *cycle_count* fields together identify a Serial Bus time that may be used for command synchronization when the *event* field specifies CYCLE_MATCH. The time identified by these fields is compared with the target's cycle clock. An equal comparison occurs if the *seconds* field matches the target's BUS_TIME register and if the *cycle_count* field matches the field of the same name in the target's CYCLE_TIME register.

The *event* field, in combination with the *sy* field or the *seconds*, and *cycle_count* fields, identifies an externally observable Serial Bus event or time that may be used as a synchronization point for the action described by the *command_block* field. Whether or not the command makes use of the event or time identified by these fields is not specified by this standard.

Value	**Event code**
0	IMMEDIATE
1	CYCLE MATCH
2	SY MATCH
3	FIRST DATA
$4 – F_{16}$	Reserved for future standardization

A value of IMMEDIATE instructs the device to commence the action indicated by the *command_block* as soon as possible.

A value of CYCLE MATCH instructs the device to commence the action indicated by the *command_block* at the cycle time specified by *seconds* and *cycle_count*.

A value of SY MATCH instructs the device to commence the action indicated by the *command_block* on the isochronous period for which the *sy* field of an isochronous packet for any enabled channel matches the *sy* field in the stream control ORB. A *event* value of SY MATCH is valid only if *talker* is zero.

A value of FIRST DATA instructs the logical unit's stream controller to perform the specified action when isochronous data is observed for any enabled isochronous channel. A *event* value of FIRST DATA is valid only if *talker* is zero.

NOTE – A *event* field value of FIRST DATA may have effects similar to IMMEDIATE, in that it is possible for isochronous data to be recorded immediately. The difference between the two stream events is apparent if no isochronous packets for any of the enabled channels are present when the stream command block ORB is executed. If IMMEDIATE is specified, CYCLE MARK packets are recorded as each cycle start is observed. If FIRST DATA is specified, no packets are recorded until the first isochronous packet for an enabled channel is observed. When this event occurs, a CYCLE MARK packet with the most recent cycle start data is recorded followed by a DATA packet for the enabled channel.

The *sy* field is valid only if *talker* is zero and the *event* field specifies SY MATCH. See the preceding description of *event*.

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

5.1.3 Stream control ORB

Stream control ORBs direct the action of a logical unit stream controller. The stream controller is configured at the time of a create stream request as either a talker or a listener. When listening, the stream controller accepts isochronous data from Serial Bus in accordance with stream control ORBs, transforms the isochronous stream and then records the data on the medium as specified by stream command block ORBs. When talking, this process is reversed and an isochronous data stream obtained from the medium is filtered and transformed by the stream controller before isochronous packets are transmitted on Serial Bus.

The format of the stream control ORB is illustrated below.

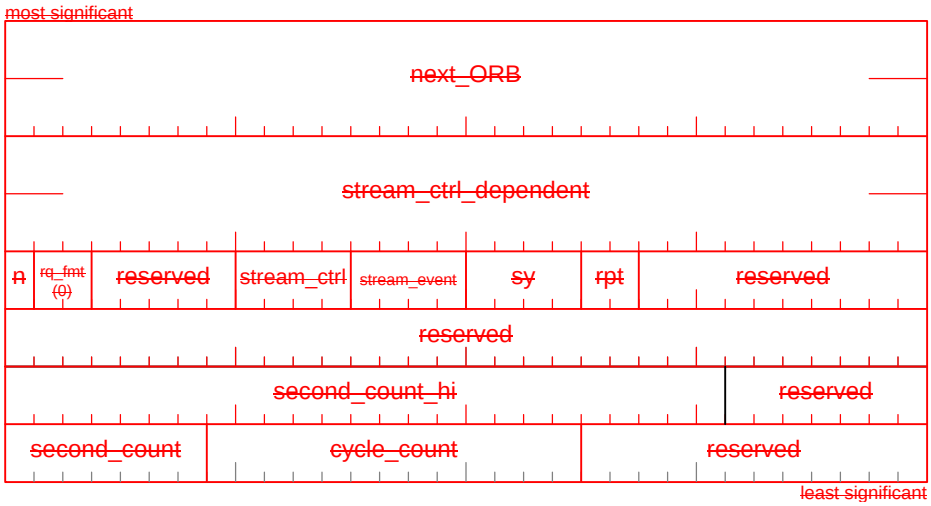

Figure 19 – Stream control ORB

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

The usage of the *stream_ctrl_dependent* field varies according to the value of *stream_ctrl* and is described in more detail for each stream control function.

The *notify* bit and *rq_fmt* field are as previously defined for all ORB formats. The *rq_fmt* field shall be zero.

The *stream_ctrl* field shall specify a stream control function for the stream, as encoded below.

Value	Stream control function
0	Reserved; not to be used
1	START
2	STOP
3	PAUSE
4	UPDATE CHANNEL MASK
5	CONFIGURE CHANNELS
6	SET ERROR MODE
7	QUERY STREAM STATUS
8 – F_{16}	Reserved for future standardization

Individual stream control functions are described in 5.1.3.1 through 5.1.3.7 below.

The *stream_event* field is valid only if the *stream_ctrl* field specifies a value of START, STOP, PAUSE or UPDATE_CHANNEL_MASK. When one of these control functions is specified, the *stream_event* field specifies the time at which the action is to take place, as encoded below.

Value	Stream event code
0	IMMEDIATE
1	CYCLE_MATCH
2	SY_MATCH
3	FIRST_DATA
4 – F_{16}	Reserved for future standardization

A value of IMMEDIATE instructs the logical unit's stream controller to perform the specified action as soon as possible, within the capabilities of the target implementation.

A value of CYCLE_MATCH instructs the logical unit's stream controller to perform the specified action at the cycle time specified by *second_count_hi*, *second_count* and *cycle_count*.

A value of SY_MATCH instructs the logical unit's stream controller to perform the specified action on the isochronous period for which the *sy* field of an isochronous packet for any enabled channel matches the *sy* field in the stream control ORB. A *stream_event* value of SY_MATCH is valid only if the logical unit's stream controller is configured as a listener.

A value of FIRST_DATA instructs the logical unit's stream controller to perform the specified action when isochronous data is observed for any enabled isochronous channel. A *stream_event* value of FIRST_DATA is valid only if the logical unit's stream controller is configured as a listener.

NOTE – A *stream_event* field value of FIRST_DATA may have effects similar to IMMEDIATE, in that it is possible for isochronous data to be recorded immediately. The difference between the two stream events is apparent if no isochronous packets for any of the enabled channels are present when the stream control ORB is executed. If IMMEDIATE is specified, CYCLE_MARK packets are recorded as each cycle start is observed. If FIRST_DATA is specified, no packets are recorded until the first isochronous packet for an enabled channel is observed. When this event occurs, a CYCLE_MARK packet with the most recent cycle start data is recorded followed by a DATA packet for the enabled channel.

The *sy* field is valid only if the logical unit's stream controller is configured as a listener and the *stream_event* field specifies SY_MATCH. See the preceding description of *stream_event*.

The *rpt* field is described in 5.1.3.6 below.

The *second_count_hi*, *second_count* and *cycle_count* fields are valid only if the *stream_ctrl* field specifies START, STOP, PAUSE or UPDATE CHANNEL MASK and the *stream_event* field specifies CYCLE MATCH. Together, these fields specify a cycle time for comparison with the target's cycle clock. An equal comparison occurs if the *second_count_hi* field matches the field of the same name in the target's BUS_TIME register and if both the *second_count* and *cycle_count* fields match their corresponding fields in the target's CYCLE_TIME register.

5.1.3.1 START stream control function

The START control function instructs the logical unit's stream controller to commence (or resume) talking or listening on Serial Bus. The time at which the action is to occur shall be specified by the *stream_event* field in conjunction with other stream control ORB fields.

5.1.3.2 STOP stream control function

The STOP control function instructs the logical unit's stream controller to terminate the isochronous stream and to flush the stream buffers. The time at which the action is to occur shall be specified by the *stream_event* field in conjunction with other stream control ORB fields.

If the target had been listening, any isochronous data already received from Serial Bus shall be made available to the stream commands previously queued at the stream command block agent. Unless prevented by other errors, the device server shall transfer all flushed isochronous data to the medium in accordance with the queued stream commands. Completion status for the stream command that completes the transfer of isochronous data shall be stored at the initiator's *status_FIFO* and shall reflect the length of the data transfer. Unexecuted stream commands that remain in the task set shall be aborted. If the target had been talking, the stream command task set shall be aborted and any untransmitted isochronous data obtained from the stream commands shall be discarded. The state of the stream command fetch agent is unaffected by the STOP control function.

5.1.3.3 PAUSE stream control function

The PAUSE control function instructs the logical unit's stream controller to suspend the transfer of isochronous data with the expectation that isochronous data transfer will resume. If the target is a talker, the stream controller shall pause on the requested stream event and shall not send any isochronous packets for the stream while paused. Subject to target implementation limitations, data from stream commands previously queued at the stream command block agent may continue to accumulate at the target while the data stream is paused. If the target is a listener, the target shall pause on the requested stream event and shall discard any isochronous packets for the stream while paused. The target may flush any isochronous data already received from Serial Bus in order to make it available to any stream commands previously queued at the stream command block agent.

5.1.3.4 UPDATE CHANNEL MASK stream control function

The UPDATE CHANNEL MASK control function instructs the logical unit's stream controller to change the set of enabled channels. The enabled channels shall be specified by the *channel_mask* field. The time at which the action is to occur shall be specified by the *stream_event* field in conjunction with other stream control ORB fields. When *stream_ctrl* specifies a value of UPDATE CHANNEL MASK, the *stream_ctrl_dependent* field shall contain a 64-bit channel mask, as shown below.

Figure 20 – Channel mask

A one in the bit position that corresponds to one of the numbered Serial Bus isochronous channels, zero to 63, indicates that the channel is to be enabled. Channel zero is represented by the most significant bit while the least significant bit represents channel 63. When a channel is enabled for listening, isochronous packets observed for that channel are transferred to the device medium under control of stream command block ORBs for the stream. Conversely, when a channel is enabled for talking, an isochronous stream is obtained from the medium as directed by stream command block ORBs and isochronous packets are transmitted on Serial Bus for the enabled channel. The channel number specified is the channel number prior to any transformation that is a result of values in the logical unit's stream controller channel map.

5.1.3.5 CONFIGURE CHANNELS stream control function

The CONFIGURE CHANNELS control function instructs the logical unit's stream controller to update the 64-entry channel map maintained internally for the isochronous stream. When listening, channel numbers observed in Serial Bus isochronous packets are replaced with numbers specified by the channel map before the isochronous data is recorded on the medium. When talking, channel numbers encountered in recorded isochronous data are replaced with numbers specified by the channel map before the isochronous packets are transmitted on Serial Bus.

NOTE – It is possible for a mapping of two or more source channels into a single destination channel to be meaningful. For example, isochronous data recorded at different times from different channels may be concatenated on the medium and subsequently replayed as a single channel.

When *stream_ctrl* specifies a value of CONFIGURE CHANNELS, the *stream_ctrl_dependent* field shall contain the address of 64-entry channel configuration map. In this case the *stream_ctrl_dependent* field shall conform to the format for address pointers specified by Figure 11 and shall address the same node as the initiator; consequently the *node_ID* field of this address pointer is reserved.

The channel configuration map consists of 64 quadlet entries. The channel configuration map is indexed by the source channel. When the target is a listener, the source channel is that observed in the isochronous packet header on Serial Bus. When the target is a talker, the source channel is that previously recorded on the medium. The format of the channel configuration map entries is illustrated below.

Figure 21 – Channel configuration map entry

The *spd* field is valid only if the target is configured as a talker. In this case *spd* determines the speed at which the isochronous packets shall be transmitted, as encoded by Table 1.

The *dest_channel* field shall specify the channel number transformation for either listening or talking. When the target is a listener, the observed channel number in the isochronous packet header shall be replaced with *dest_channel* as the data is recorded on the medium. When the target is a talker, the channel number obtained from the medium shall be replaced with *dest_channel* at the time the isochronous packet is transmitted.

The *sid* field is valid only if the target is configured as a talker. If the recorded isochronous data conforms to the CIP format described in Annex G, the *sid* field in the previously recorded CIP header shall be replaced with the *sid* value specified by the channel map when the packet is transmitted.

5.1.3.6 SET ERROR MODE stream control function

The SET ERROR MODE control function instructs the logical unit's stream controller to configure its error handling mode as specified by the *rpt* field.

The *rpt* field specifies an operational mode for the logical unit's stream controller, as described in the table below. The *rpt* field is valid only if the *stream_ctrl* field specifies SET ERROR MODE.

Value	Error handling mode
0	Report errors and halt stream
1	Report errors and continue stream
2	Ignore all errors
3	Reserved for future standardization

Different sorts of errors may be detected when the logical unit's stream controller is configured as a talker or a listener. If an error occurs, the stream controller shall take one of three actions, as specified by the value of *rpt*:

- Report the error by writing unsolicited status to the initiator and then halting isochronous data transfers by performing the equivalent of a STOP control function with a *stream_event* value of IMMEDIATE;
- Report the error by writing unsolicited status to the initiator but continue isochronous data transfers; or
- Ignore the error and continue isochronous data transfers.

A more detailed description of isochronous errors and how they are handled is provided in 12.3.

5.1.3.7 QUERY STREAM STATUS stream control function

The QUERY STREAM STATUS control function instructs the logical unit's stream controller to return status information that indicates whether or not the stream controller is ready to accept a START control function.

NOTE – Assume that a target is to be instructed to listen to isochronous data and transfer the stream to device medium. If the starting medium location is at a nonzero offset relative to a block boundary, some implementations may require time to read previously recorded data from the medium before being ready to commence recording the new isochronous data. Subsequent to queuing a stream command ORB at the stream command block agent, the QUERY STREAM STATUS control function may be used to determine if the target is ready to accept a START control function.

5.1.4 Management ORBs

Table 2 – Management request functions

Value	Mandatory	Management function
0	Yes	LOGIN
1	Yes	QUERY LOGINS
2		CREATE STREAM
3	Yes	RECONNECT
4		SET PASSWORD (see Annex C)
5		NODE HANDLE
6		~~Reserved for future standardization~~ STREAM CONTROL
7	Yes	LOGOUT
8 – A_{16}		Reserved for future standardization
B_{16}		ABORT TASK
C_{16}	Yes	ABORT TASK SET
D_{16}		Reserved for future standardization
E_{16}	Yes	LOGICAL UNIT RESET
F_{16}	Yes	TARGET RESET

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.

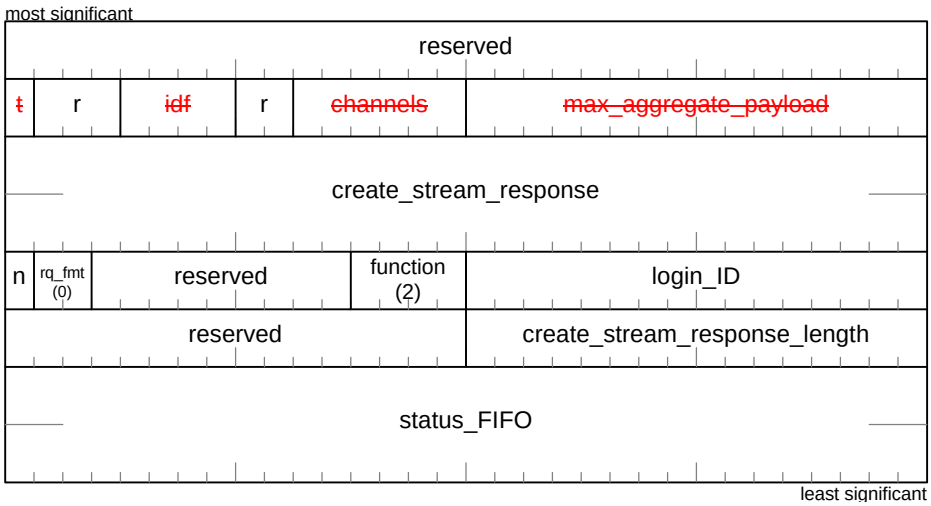

Figure 27 – Create stream ORB

~~The *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 *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 *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.

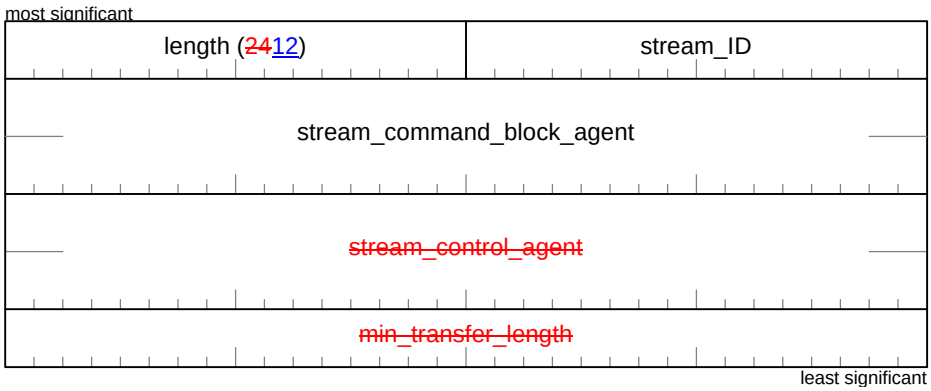

Figure 28 – Create stream response

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.~~

5.1.4.5a Stream control ORB

The stream control ORB permits characteristics of a target stream controller to be interrogated or established. The format of this management ORB is shown below.

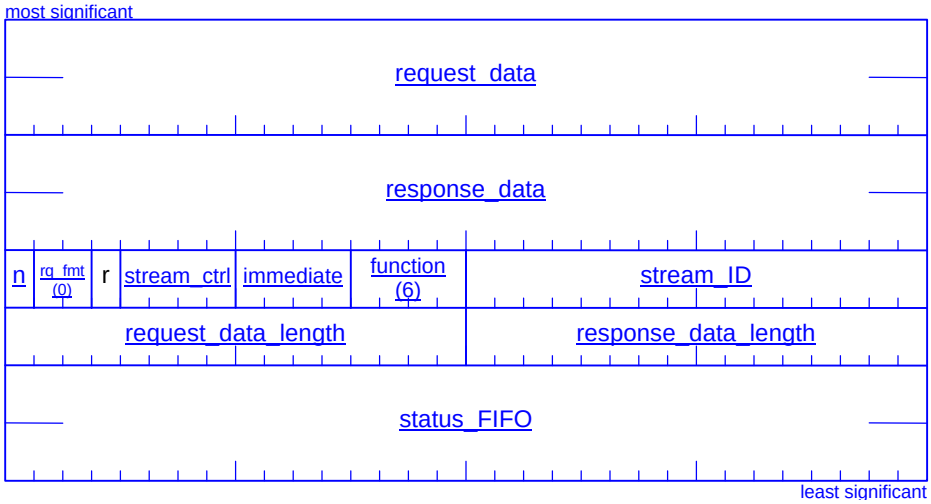

Figure 31a – Stream control ORB

The *request_data* and *request_data_length* fields specify the address and size, in bytes, of a buffer allocated for optional request data pertinent to the stream control operation. The *request_data* field shall conform to the format for address pointers specified by Figure 11 and shall address the same node as the initiator; consequently the *node_ID* field of this address pointer is reserved. The buffer shall be accessible to a Serial Bus block read request with a data transfer length less than or equal to *request_data_length*.

The *response_data* and *response_data_length* fields specify the address and size, in bytes, of a buffer allocated for the return of optional response data that results from the stream control operation. The *response_data* field shall conform to the format for address pointers specified by Figure 11 and shall address the same node as the initiator; consequently the *node_ID* field of this address pointer is reserved. The buffer shall be accessible to a Serial Bus block write request with a data transfer length less than or equal to *response_data_length*.

Both stream control request and response data use the same basic format, illustrated by Figure 31b below.

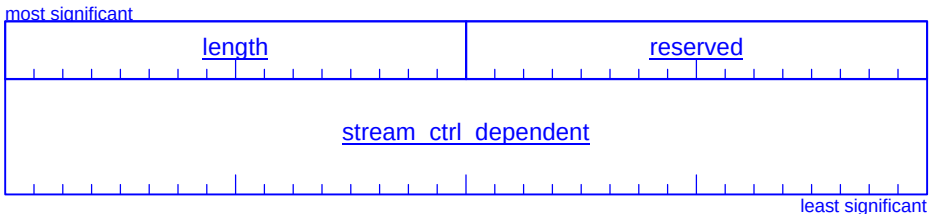

Figure 31b – Stream control request / response buffer format

The *length* field shall contain the length, in bytes, of the request or response data. If the stream control operation fails, the contents of the response buffer are unspecified. Otherwise, upon successful completion of the stream control operation, the target shall store response data up to the length specified

by *response_data_length* and set the *length* field in the response data to the number of bytes actually stored.

The *notify* bit, *rq_fmt* and *status_FIFO* fields are as previously defined for management ORB formats.

The *stream_ctrl* field shall specify a stream control function for the stream, as encoded below.

Value	Stream control function
0	Reserved; not to be used
1	CONFIGURE CHANNELS
2	SET ERROR MODE
3 – F_{16}	Reserved for future standardization

Individual stream control functions are described in 5.1.4.5a.1 through 5.1.4.5a.2 below.

The *immediate* field permits *stream_ctrl*-dependent information to be provided in the ORB in lieu of or in addition to data in the request buffer.

The *stream_ID* shall be set to a value returned in create stream response data.

5.1.4.5a.1 CONFIGURE CHANNELS stream control function

The CONFIGURE CHANNELS control function instructs the stream controller to update its internal 64-entry channel map with the information provided in the request buffer. When listening, channel numbers observed in Serial Bus isochronous packets are replaced with numbers specified by the channel map before the isochronous data is recorded on the medium. When talking, channel numbers encountered in recorded isochronous data are replaced with numbers specified by the channel map before the isochronous packets are transmitted on Serial Bus.

NOTE – It is possible for a mapping of two or more source channels into a single destination channel to be meaningful. For example, isochronous data recorded at different times from different channels may be concatenated on the medium and subsequently replayed as a single channel.

When *stream_ctrl* specifies a value of CONFIGURE CHANNELS, the *stream_ctrl_dependent* field shall contain one or more entries that conform to the format illustrated by Figure 18. For entry provided, the stream controller shall update its internal channel map. The meanings of the fields in each entry are as previously specified in 5.1.2.2.

5.1.4.5a.2 SET ERROR MODE stream control function

The SET ERROR MODE control function instructs the stream controller to configure its error handling mode as specified by the *immediate* field, which specifies an operational mode for the stream controller, as described in the table below.

Value	Error handling mode
0	Report errors and halt stream
1	Report errors and continue stream
2	Ignore all errors
3	Reserved for future standardization

Different sorts of errors may be detected when the logical unit's stream controller is configured as a talker or a listener. If an error occurs, the stream controller shall take one of three actions, as specified by the value of *immediate*:

- Report the error by writing unsolicited status to the initiator and then immediately halting isochronous data transfers;
- Report the error by writing unsolicited status to the initiator but continue isochronous data transfers; or
- Ignore the error and continue isochronous data transfers.

A more detailed description of isochronous errors and how they are handled is provided in 12.3.