TA Document 1999029

AV/C Disc Subunit Enhancements for Hard Disk Drive Specification

July 10, 2000

Sponsored by:
1394 Trade Association

Accepted for Release by:
1394 Trade Association Board of Directors.

Abstract:
This specification defines a set of enhancements to the AV/C Disc Subunit General Model Specification to provide definition for video data types and provide enhanced support for AV/C HDD Devices.

Keywords:
Audio, Video, 1394, Digital, Interface, Storage.

1394 Trade Association Specifications are developed within Working Groups of the 1394 Trade Association, a non-profit industry association devoted to the promotion of and growth of the market for IEEE 1394-compliant products. Participants in working groups serve voluntarily and without compensation from the Trade Association. Most participants represent member organizations of the 1394 Trade Association. The specifications developed within the working groups represent a consensus of the expertise represented by the participants.

Use of a 1394 Trade Association Specification is wholly voluntary. The existence of a 1394 Trade Association Specification is not meant to imply that there are not other ways to produce, test, measure, purchase, market or provide other goods and services related to the scope of the 1394 Trade Association Specification. Furthermore, the viewpoint expressed at the time a specification is accepted and issued is subject to change brought about through developments in the state of the art and comments received from users of the specification. Users are cautioned to check to determine that they have the latest revision of any 1394 Trade Association Specification.

Comments for revision of 1394 Trade Association Specifications are welcome from any interested party, regardless of membership affiliation with the 1394 Trade Association. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments.

Interpretations: Occasionally, questions may arise about the meaning of specifications in relationship to specific applications. When the need for interpretations is brought to the attention of the 1394 Trade Association, the Association will initiate action to prepare appropriate responses.

Comments on specifications and requests for interpretations should be addressed to:

Editor, 1394 Trade Association
Regency Plaza Suite 350
2350 Mission College Blvd.
Santa Clara, Calif. 95054, USA

1394 Trade Association Specifications are adopted by the 1394 Trade Association without regard to patents which may exist on articles, materials or processes or to other proprietary intellectual property which may exist within a specification. Adoption of a specification by the 1394 Trade Association does not assume any liability to any patent owner or any obligation whatsoever to those parties who rely on the specification documents. Readers of this document are advised to make an independent determination regarding the existence of intellectual property rights, which may be infringed by conformance to this specification.

Table of Contents

List of Figures

List of Tables

1. Overview

1.1 Purpose

This specification defines enhancements to the AV/C Disc General Model Specification.

1.2 Scope

The enhancements in this document provide definition for video data types and enhance support in the Disc Subunit General Model for managing the storage and retrieval of AV content for HDD Player/Recorder devices.

2. References

The following standards contain provisions, which through reference in this document constitute provisions of this standard. All the standards listed are normative references. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below.

[R1] ISO/IEC 13123:1994, Control and Status Register (CSR) Architecture for Microcomputer Buses[1]

[R2] IEEE Std 1394-1995, Standard for a High Performance Serial Bus[2]

[R3] IEC 61883, Digital Interface for Consumer Electronic Audio/Video Equipment[3]

[R4] AV/C Digital Interface Command Set General Specification, version 3.0[4]

[R5] AV/C Compatible Asynchronous Serial Bus Connections[4]

[R6] AV/C Commands for Management of Asynchronous Serial Bus Connections[4]

[R7] AV/C Disc Subunit Model and Command Set[4]

[R8] Enhancements to the AV/C General Specification 3.0[4]

[1] ISO/IEC [ANSI/IEEE] publications are available from the Institute of Electrical and Electronics Engineers, Service Center, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA or from International Electrotechnical Commission, 3 rue de Varembé, Case Postale 131, CH 1211, Genève 20, Switzerland/Suisse.

[2] IEEE publications are available from the Institute of Electrical and Electronics Engineers, Service Center, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331 USA.

[3] Document 100C/182/FDIS from U.S. National Committee of The IEC ANSI, 11 West 42nd Street, 13th Floor, New York, NY 10036 USA. Phone: +1.212.642.4900 (Questions) 212.642.4980 (Sales). FAX: +1.212.398.0023.

[4] Accepted 1394 Trade Association (AV/C) documents may be ordered through 1394 Trade Association, Regency Plaza Suite 350, 2350 Mission College Blvd., Santa Clara, Calif. 95054, USA; by contacting taadmin@1394ta.org; by retrieving a PDF-format copy from the 1394 Trade Association web-site at: http://www.1394ta.org/Technology/Specifications/index.htm.

3. Definitions

3.1 Conformance levels

3.1.1 expected: A key word used to describe the behavior of the hardware or software in the design models *assumed* by this Specification. Other hardware and software design models may also be implemented.

3.1.2 may: A key word that indicates flexibility of choice with *no implied preference*.

3.1.3 shall: A key word indicating a mandatory requirement. Designers are *required* to implement all such mandatory requirements.

3.1.4 should: A key word indicating flexibility of choice with a strongly preferred alternative. Equivalent to the phrase *is recommended*.

3.1.5 reserved fields: A set of bits within a data structure that are defined in this specification as reserved, and are not otherwise used. Implementations of this specification shall zero these fields. Future revisions of this specification, however, may define their usage.

3.1.6 reserved values: A set of values for a field that are defined in this specification as reserved, and are not otherwise used. Implementations of this specification shall not generate these values for the field. Future revisions of this specification, however, may define their usage.

3.2 Glossary of terms

3.2.1 AV Content Object: An Object Entry that describes an AV Track and its contents. This term applies to Digital Video, Audio, Digital Still Images, among others. AV Content Objects have an *entry_type* value in the range of 80_{16} to $8F_{16}$.

3.2.2 AV Frame: A portion of an AV Track defined by the specific data encoding method applied to that AV Track. For example, MPEG encoded AV Tracks have I, P, and B frames as defined by MPEG documentation.

3.2.3 AV Segment: The portion of an AV Track between two adjacent positions. AV segments are useful for performance and editing operations on AV Tracks.

3.2.4 AV subunit: an instantiation of a virtual entity that can be identified uniquely within an AV unit and offers a set of coherent functions.

3.2.5 AV Track: A collection of recorded data described by an AV Content Object Entry. AV Tracks include video, audio, textual, and other types of data. Although sometimes referred to as “Tracks,” the name “AV tracks” is intended to differentiate AV content from other tracks normally associated with hard disk drive devices.

3.2.6 AV/C: Audio/Video Control, as in the AV/C Digital Interface Command Set specified by this document.

3.2.7 byte: Eight bits of data.

3.2.8 CSR: Control and Status Register, a term defined by IEEE Std 1394-1995.

3.2.9 EUI-64: Extended Unique Identifier, 64 bits, as defined by the IEEE. The EUI-64 is a concatenation of the 24-bit company_ID obtained from the IEEE Registration Authority Committee (RAC) and a unique 40-bit number (typically a silicon serial number) that the vendor supplies by company_ID. The EUI-64 is also known as the node unique ID and is redundantly present in a node's configuration ROM in both the Bus_Info_Block and the Node_Unique_Id leaf.

3.2.10 FCP: Function Control Protocol, term defined by IEC 61883, Digital Interface for Consumer Electronic Audio/Video Equipment.

3.2.11 IEEE: The Institute of Electrical and Electronics Engineers, Inc.

3.2.12 isochronous: A term that indicates the essential characteristic of a time-scale or signal, such that the time intervals between consecutive instances either have the same duration or durations that are integral multiples of the shortest duration. In the context of Serial Bus, "isochronous" is taken to mean a bounded worst-case latency for the transmission of data; physical and logical constraints that introduce jitter preclude the exact definition of "isochronous."

3.2.13 module: The smallest component of physical management, *i.e.*, a replaceable device.

3.2.14 nibble: Four bits of data. A byte is composed of two nibbles.

3.2.15 node: An addressable device attached to Serial Bus with at least the minimum set of control registers defined by IEEE Std 1394–1995.

3.2.16 node ID: A 16-bit number, unique within the context of an interconnected group of Serial Buses. The node ID is used to identify both the source and destination of Serial Bus asynchronous data packets. It can identify one single device within the addressable group of Serial Buses (unicast), or it can identify all devices (broadcast).

3.2.17 Object Entry: AV/C descriptor structure type that contains descriptive information. It is commonly used to describe AV-content (a track) that is recorded and stored on the disc subunit media.

3.2.18 Object List: AV/C descriptor structure type that contains some number of Object Entries.

3.2.19 PCR: Plug Control Register, a term defined by IEC 61883, Digital Interface for Consumer Electronic Audio/Video Equipment.

3.2.20 iPCR: Input plug PCR, as defined by IEC 61883.

3.2.21 oPCR: Output plug PCR, as defined by IEC 61883.

3.2.22 plug: A physical or virtual end-point of connection implemented by an AV unit or by a subunit that may receive or transmit isochronous data or other data. Plugs may be Serial Bus plugs, accessible through the PCRs; plugs may also be external physical plugs on the AV unit; or they may be internal virtual plugs implemented by the AV subunits.

3.2.23 quadlet: Four bytes of data.

3.2.24 Serial Bus: The physical interconnections and higher level protocols for the peer-to-peer transport of serial data, as defined by IEEE Std 1394–1995.

3.2.25 stream: A time-ordered set of digital data originating from one source and terminating at zero or more sinks. A stream is characterized by bounded bandwidth requirements and by synchronization points, or time stamps, within the stream data.

4. Enhanced Definitions

The HDD device has many unique capabilities. One unique capability of the HDD device is its large storage capacity, which allows the implementation of many advanced features. A relatively large number of AV tracks (AV content) and associated structures may be stored in the HDD device by the user. These AV tracks can be of several different types including audio, video, digital still image, and text. All of these AV tracks can be encoded using methods such as MPEG and DVCR (DVC).

Given the large number of AV tracks, data types, and possible encoding formats, it is important to provide the user with a robust HDD model for describing, categorizing, and sorting these tracks in the HDD device. The HDD model should support multiple users such as the members of a household, and provide the users with the ability to share AV content safely and reliably.

For these reasons, enhancements to the AV/C Disc Subunit Model are proposed to allow a reliable, consistent, and user-definable hierarchy of descriptive information to aid the categorization and the organization of a large number of AV content and their descriptors. This allows the user to interact with a larger number of AV content easily.

4.1 AV Track

Recorded AV content is also called an AV track. Examples of AV tracks are recorded songs or movies. In general, an AV track is a sequence of recorded data and is described by an AV/C object entry. An AV track has two endpoints: the beginning and the end of the AV track as shown in Figure 4.1 – Generic AV track.

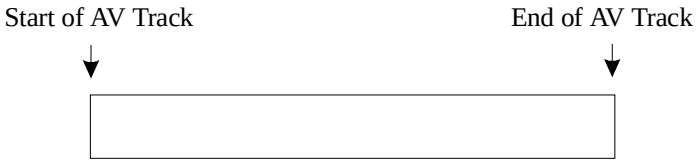

Figure 4.1 – Generic AV track

Typically, an AV track is recorded or played from beginning to end.

4.2 AV Frame

An AV frame is a uniquely identifiable section of an AV track, as illustrated in Figure 4.2 – Generic AV frames. Identification of AV frames depend on the encoding method used for the data contained in the AV track. For example, if an AV track were encoded using MPEG compression, then this AV track would have I, P, and B frames as defined by MPEG documentation. If an AV track were encoded using DVC compression, then this AV track would have AV frames as define in the DVC documentation.

Although AV frames are identified by encoding formats, not necessarily all encoding formats are supported by every disc device. Therefore, AV frames may not be supported by a disc device if the encoding method were not supported by that device. Hence, command operations based on AV frames may not be supported by the device. It is a matter of implementation choice whether or not a disc device supports AV frames. When a disc device supports AV frames, the definition of an AV frame is data encoding specific (e.g. MPEG or DVCR format).

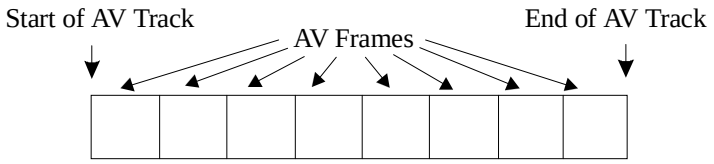

Figure 4.2 – Generic AV frames

4.3 AV Segment

AV tracks can be separated into multiple sections called AV segments. AV segments are uniquely identifiable sections of an AV track and can be specified as parameters in some AV commands.

An AV segment is the section of an AV track between two adjacent segment marker positions. These segment marker positions are described in section 5.2.

When a disc subunit supports AV Frames, all segment marker positions are expected to align exactly on a frame boundary depending on the data-encoding format.

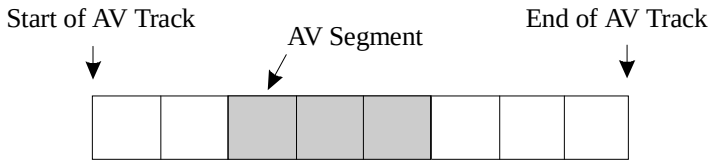

Figure 4.3 – Generic AV segment

The position markers that specify this AV segment are not illustrated in the figure above because they are not part of the AV track. They are kept in the AV content object for this track and are described in section 5.1.

5. Content Description

This section contains definitions that are appropriate to the AV/C Disc General Model. If some or all of these definitions are more appropriate to other AV/C documents, they can be placed there later. The following table shall be used to extend the current list of defined info block structures.

Table 5.1 – New info block type values

Info_block_type	Name
$880C_{16}$	Position_marker_info_block
$880D_{16}$	Segment_marker_info_block
$880E_{16}$	Record_position_info_block
$880F_{16}$	Playback_position_info_block
8810_{16}	Latest_record_mode_info_block
8811_{16}	Previously_recorded_segment_info_block
8812_{16}	Video_signal_mode_info_block
8813_{16}	Video_stream_format_info_block
8814_{16}	Program_attribute_info_block
8815_{16}	Object_subtype_info_block

5.1 Position Marker Info Block ($880C_{16}$)

Position markers are very similar to position indicator defined in Enhancement to the AV/C General Specification 3.0, section 6.3, with one essential difference: position *indicators* can change spontaneously whereas position *markers* cannot. Position indicators are either reported as status information and subject to spontaneous changes, or they are provided as parameters to commands and expire after command execution has been completed.

Unlike position indicators, position markers have a persistent quality. Position markers do not change spontaneously and must be changed explicitly by command. Because position markers are stored as info block structures in the object entry that describes the AV track, they do not expire after command execution has been completed.

Address	Length, bytes	Contents
00 00_{16}	2	Compound_length
00 01_{16}		
00 02_{16}	2	Info_block_type = (position marker)
00 03_{16}		
00 04_{16}	2	Primary_fields_length
00 05_{16}		
00 06_{16}	1	Position_marker_type
00 07_{16}		Position_marker_specific_information
:		
:		

Figure 5.1 – Position marker info block

The *Position_marker_type* field specifies what type of position marker entry is contained in the *Position_marker_specific_information* field. Position *marker* entry types have the same format as similarly named position *indicator* entries. The position marker entry types are defined below.

Table 5.2 – Position marker type values

value	Meaning
00 00_{16}	relative_HMSF_count
00 01_{16}	relative_segment_count
00 02_{16}	Reserved
00 03_{16}	relative_byte_count
00 04_{16}	Reserved
00 05_{16}	Reserved
00 06_{16}	Reserved
00 07_{16}	Reserved
All other values	Reserved

The *Position_marker_specific_information* field contains a position marker entry. Position marker entries for this info block have the same format as similarly named position indicators as specified by the marker type field.

NOTE —Edit_location_indicators have a better mapping of values for absolute and relative types. It is not clear why this is not true for position_indicator types.

Position markers can be further described by using nested info block structures.

5.2 Segment Marker Info Block ($880D_{16}$)

Segment markers are stored in object entry-specific area of the AV content object that describes a segmented AV track. Segment markers are very similar to position markers with one essential difference: a *position* marker specifies a single location within an AV track, whereas a *segment* marker specifies the position of a segment boundary in an AV track. Position marker info blocks specify a single position within an AV Track. A segment marker info block contains an ordered list of segment marker entries that define AV segment boundaries.

The structure of a segment marker info block is similar to the position marker info block. The segment marker info block is defined below.

Address	Length, bytes	Contents
$00\ 00_{16}$	2	Compound_length
$00\ 01_{16}$		
$00\ 02_{16}$	2	Info_block_type = (segment marker)
$00\ 03_{16}$		
$00\ 04_{16}$	2	Primary_fields_length
$00\ 05_{16}$		
$00\ 06_{16}$	1	Segment_marker_type
$00\ 07_{16}$	1	number_of_segment_marker_entries[n]
$00\ 08_{16}$		Segment_marker_entry[0]
:		
:		
:		:
:		
:		
:		Segment_marker_entry[n-1]
:		
:		

Figure 5.2 – Segment marker info block

The *Segment_marker_type* field specifies what type of segment marker entries are contained in this segment marker info block. All segment marker entries in the same block have the same segment marker type. Segment marker entry types have the same format as similarly named position marker entries described in Table 5.2.

The *number_of_segment_marker_entries* field specifies the number of segment marker entries found in the segment marker info block.

Address	Length, bytes	Contents
00 00_{16}		Segment_marker_specific_information
:		
:		

Figure 5.3 – Segment marker entry

The *Segment_marker_specific_information* field contains a *position_indicator_part*. Segment marker entries for this info block have the same format as similarly named position markers specified by the segment marker type field as described in Table 5.2.

All segment marker entries shall be stored in increasing order such that any segment marker entry[i] is closer to the start of track than segment marker entry[i+1]. Because segment marker entries are in increasing order, the *start_position* of any segment[i] is found in segment marker entry[i-1] except as stated below. For example, The *start_position* of segment[7] is found in segment marker entry[6] and, the *end_position* of segment[7] is found in segment marker entry[7].

The start of track is the implied *start_position* for segment 0. Otherwise, all segment boundaries must be specified. The *start_position* of segment[0] is the start of the track and, the *end_position* of segment[0] is found in segment marker entry[0].

When any segment[i] is specified as a command operand, segment[i] shall specify the segment bounded by segment marker entry[i-1] and segment marker entry[i]. A segmented AV track is illustrated in below.

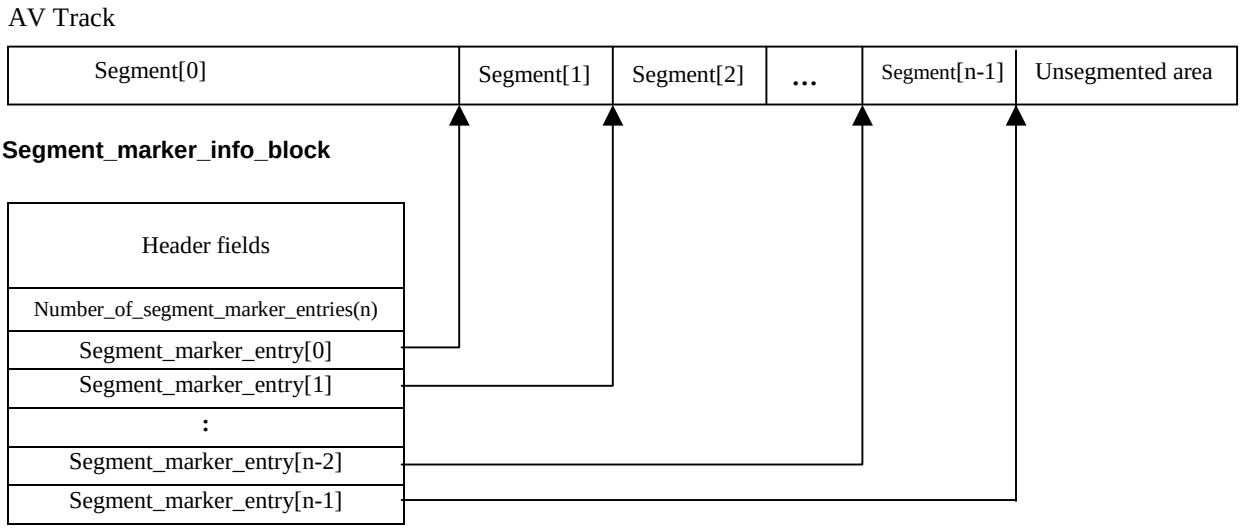

Figure 5.4 – Example of segmented AV track

5.3 Record Position Info Block ($880E_{16}$)

This info block is useful for memorizing the last used recording position of this AV Track by placing it into the AV Content Object entry.

Address	Length, bytes	Contents
$00\ 00_{16}$	2	Compound_length
$00\ 01_{16}$		
$00\ 02_{16}$	2	Info_block_type = (record position)
$00\ 03_{16}$		
$00\ 04_{16}$	2	Primary_fields_length
$00\ 05_{16}$		
$00\ 06_{16}$	1	Position_marker_type
$00\ 07_{16}$: :		Position_marker_specific_information

Figure 5.5 – Record position info block

The *Position_marker_type* field specifies what type of position marker entry is contained in the *Position_marker_specific_information* field. Position *marker* entry types have the same format as similarly named position *indicator* entries.

5.4 Playback Position Info Block ($880F_{16}$)

This info block is useful for memorizing the last used playback position of this AV Track by placing it into the AV Content Object entry.

Address	Length, bytes	Contents
00 00_{16}	2	Compound_length
00 01_{16}		
00 02_{16}	2	Info_block_type = (playback position)
00 03_{16}		
00 04_{16}	2	Primary_fields_length
00 05_{16}		
00 06_{16}	1	Position_marker_type
00 07_{16} : :		Position_marker_specific_information

Figure 5.6 – Playback position info block

The *Position_marker_type* field specifies what type of position marker entry is contained in the *Position_marker_specific_information* field. Position *marker* entry types have the same format as similarly named position *indicator* entries. The position marker entry types are defined in section 5.1.

5.5 Previously Recorded Segment Info Block (8811_{16})

This info block is useful for the controller to determine which area of the AV Track has previously been recorded. The previously recorded area is described as an AV Segment (see section 4.3) with one boundary at the start of track and the other boundary defined by this info block. When present, this info block defines the position beyond which playback or recording shall not be allowed. Recording may begin at this position but any attempt to playback or record the AV Track beyond this position shall be rejected by the subunit.

Address	Length, bytes	Contents
00 00_{16}	2	Compound_length
00 01_{16}		
00 02_{16}	2	Info_block_type = (previously recorded segment)
00 03_{16}		
00 04_{16}	2	Primary_fields_length
00 05_{16}		
00 06_{16}	1	Position_marker_type
00 07_{16}		Position_marker_specific_information
:		
:		

Figure 5.7 – Previously recorded segment info block

The *Position_marker_type* field specifies what type of position marker entry is contained in the *Position_marker_specific_information* field. Position *marker* entry types have the same format as similarly named position *indicator* entries. The position marker entry types are defined in section 5.1.

5.6 Latest Recording Mode Info Block (8810_{16})

This info block is useful for memorizing the last recording method used to modify this AV Track by placing it into the AV Content Object entry associated with this AV Track.

Address	Length, bytes	Contents
00 00_{16}	2	Compound_length
00 01_{16}		
00 02_{16}	2	Info_block_type = (latest recording mode)
00 03_{16}		
00 04_{16}	2	Primary_fields_length
00 05_{16}		
00 06_{16}	1	Recording_mode

Figure 5.8 – Latest recording mode info block

The currently defined recording mode value useful for this info block as shown below.

Table 5.3 – Recording_mode values

value	meaning
01_{16}	Object was used for loop recording
All others	Reserved for future specification

5.7 Object Subtype Info Block (8815_{16})

This info block is useful for describing extended characteristics of an AV Content Object. It is placed into the AV Content Object entry.

Address	Length, bytes	Contents
00 00_{16}	2	Compound_length
00 01_{16}		
00 02_{16}	2	Info_block_type = (object subtype)
00 03_{16}		
00 04_{16}	2	Primary_fields_length
00 05_{16}		
00 06_{16}	1	Object_creation_method

Figure 5.9 – Object subtype info block

The currently defined object creation methods are shown below.

Table 5.4 – object creation method values

value	meaning
01_{16}	Object created by RECORD command
02_{16}	Object created by RECORD OBJECT command
03_{16}	Object created by CREATE AV TRACK command
FF_{16}	No information
All others	Reserved for future specification

5.8 Video Content Object (83_{16})

This section defines Video Content Object, which is a disc subunit specific object type. Video Content Object describes AV content stored within the subunit; therefore, it is a member of AV Content Objects.

Table 5.5 – Video object entry type

entry_type	Name
83_{16}	Video content object (including data such as MPEG or DVC with optionally embedded audio, but not digital still images or plain audio which have their own *entry_type* specifications)

5.8.1 Video Object Specific Information

An AV object with *entry_type* of video content describes the video content of an AV track and has the following *entry_specific_information*.

Address	Length, bytes	Contents
00 00_{16}	2	non_info_block_fields_length
00 01_{16}		
00 02_{16}	1	disc_subunit_object_attributes
00 03_{16}		video_signal_mode_info_block
:		
:		
:		optional info blocks
:		
:		

Figure 5.10 – Video object entry_specific_information

The *video_signal_mode_info_block* specifies the parameters used for recording video content and is required.

5.9 Video Signal Mode Info Block (8812_{16})

The *video_signal_mode_info_block* contains information that describes the signal format in use during recording of a video AV Track. This info block can be placed in the AV Content Object entry to memorize the video signal mode used to create the AV Track. This info block is also useful for plug status during both recording and playback of video AV Tracks and may be nested in the plug status info block. The video signal mode info block has the following format.

Address	Length, bytes	Contents
00 00_{16}	2	Compound_length
00 01_{16}		
00 02_{16}	2	Info_block_type = (video_signal_mode_info_block)
00 03_{16}		
00 04_{16}	2	primary_fields_length
00 05_{16}		
00 06_{16}	1	video_signal_mode
00 07_{16}		Video_signal_mode_specific_information
:		
:		

Figure 5.11 – Video_signal_mode_info_block

The *video_signal_mode* specifies the type of video signal in effect. Table 5.6 specifies the defined values for this field.

Table 5.6 – Video signal mode

Value	signal_mode
00_{16}	DVCR
01_{16}	MPEG2-TS
02_{16}	DTV/Satellite-System B
all other values	Reserved for future specification

The DVCR signal modes are defined in the IEC 61883, part 2, part 3, and part 5.

The MPEG2-TS signal mode is defined in the IEC 61883, part 4.

The DTV/Satellite System, B Mode is defined by ITU-R BO.1294.

5.9.1 DVCR Signal Mode

The *video_signal_mode_specific_information* field is defined in Figure 5.12.

Address Offset	length	msb							lsb
00 00_{16}	1	DVCR_signal_format							
00 01_{16}	1	audio_ctrl [3]			audio_ctrl [2]		audio_ctrl [1]		audio_ctrl [0]
00 02_{16}	1	audio_channels [1]			sample_freq [1]		audio_channels [0]		sample_freq [0]
00 03_{16}	1	audio_channels [3]			sample_freq [3]		audio_channels [2]		sample_freq [2]

Figure 5.12 – Mode specific information for DVCR signal mode

The *DVCR_signal_format* field specifies the DVCR signal mode, which can have one of the values defined below.

Table 5.7 – DVCR signal mode – signal formats

Value	DVCR_signal_format
00_{16}	SD 525-60
04_{16}	SDL 525-60
08_{16}	HD 1125-60
80_{16}	SD 625-50
84_{16}	SDL 625-50
88_{16}	HD 1250-50
FF_{16}	No Information
All other values	Reserved for future specification

The *audio_ctrl*[n], *audio_channels*[n], *sample_freq*[n] fields have the same interpretation as the AUDIO MODE control command format in the section 4.4 of AV/C Tape Recorder/Player Subunit Specification Version 2.1.

If there is no information for DVCR signal mode, then all audio_ctrl[n], audio_channels[n] and sample_freq[n] fields shall be set to 11_2.

5.9.2 MPEG2-TS Signal Mode

The *video_signal_mode_specific_information* field for MPEG2-TS signal mode is defined below.

Address Offset	length	msb							lsb
00 00_{16}	1	frame_rate_code				Reserved			
00 01_{16}	3	(MSB)							
00 02_{16}		video_signal_bit_rate (x 10^3 bit/sec)							
00 03_{16}		(LSB)							

Figure 5.13 – Mode specific information for MPEG2-TS signal mode

The *frame_rate_code* field indicates the frame rate of MPEG2-TS video stream.

Table 5.8 – Frame rate code

Value	frames/second
00_{16}	Forbidden
01_{16}	24000/1001
02_{16}	24
03_{16}	25
04_{16}	30000/1001 (29.97)
05_{16}	30
06_{16}	50
07_{16}	60000/1001
08_{16}	60
$0F_{16}$	No information
all other values	Reserved for future specification

The *video_signal_bit_rate* field specifies the bit rate of the video signal recorded by the disc subunit. This field is encoded in BCD format and has a unit of 10^3 bit/sec. The value of FF FF FF_{16} means no information.

5.9.3 DTV/Satellite-System B Signal Mode

The *video_signal_mode_specific_information* field for DTV/Satellite-System B signal mode is defined below.

Address Offset	length	msb							lsb
00 00_{16}	1	frame_rate_code				Reserved			
00 01_{16}	3	(MSB)							
00 02_{16}		reserved							
00 03_{16}		(LSB)							

Figure 5.14 – Mode specific information for MPEG2-TS signal mode

Frame rate code values are the same as specified in Table 5.8 – Frame rate code.

5.10 Video Stream Format Subtype Info Block (8813_{16})

Video stream format subtype info block shows the subtype name of the data stream. In case a recorded program was provided through a broadcast system, this info block specifies the ID of the broadcast system. (e.g. “SKY PerfecTV!”)

NOTE —There are three public standards (ATSC, DVB, ARIB) and many other private standards (ex. SkyPerfecTV!, CANAL+, etc.) broadcasting with MPEG2-TS stream. These standards are not identical, and so a method to distinguish between stream types is required. Video_stream_format_subtype info block is designed for this purpose.

Address	Length, bytes	Contents
00 00_{16}	2	compound_length
00 01_{16}		
00 02_{16}	2	Info_block_type = (video_stream_format_subtype_info_block)
00 03_{16}		
00 04_{16}	2	Primary_fields_length
00 05_{16}		
00 06_{16}	1	Validity_flags
00 07_{16}	4	Format Identifier
:		
00 $0A_{16}$		
00 $0B_{16}$	9	Network information
:		
00 13_{16}		
00 14_{16}	16	Stream_format_name
:		
00 23_{16}		

Figure 5.15 – Video stream format subtype info block

5.10.1 Primary Fields length

The Primary_fields_length contains a 2-byte value that defines the total combined length of the validity_flags, format_identifier, network_information and stream_format_name fields as a number of bytes.

5.10.2 Validity Flags

The valid flags provide information concerning the validity of both the format_identifier, and network_information fields. According to ISO/IEC 13818-1, we should use *Format Identifier* for identification, but some broadcasting standards do not contain an effective *format identifier* value in their stream. Therefore the *Video_stream_format_subtype* info block contains both a *Format Identifier* and *Network Information* field. The *validity flags* are used to indicate the validity of these two fields.

Address Offset	length	msb						lsb
00 00_{16}	1	FI	NI	Org_network	Transport_id	Service_id	Country_code	Reserved

Figure 5.16 – Validity_flags

When set to 1, the FI is a flag indicates that the *Format Identifier* field contains valid information

When set to 1, the NI is a flag indicates that the *Network Information* field contains valid information

When set to 1, the Org_network, Transport_id, Service_id and country_code flags indicate the validity of corresponding data in Network Information field. If the corresponding data is not valid, or if the NI flag is set to 0, then these flags shall also be set to 0.

When set to 1, the Org_network flag indicates the validity of Original Network ID data in the Network Information, field.

When set to 1, the Transport_id flag indicates the validity of Transport Stream ID data in the Network Information, field.

When set to 1, the Service_id flag indicates the validity of Service ID data in the Network Information, field.

When set to 1, the Country_code flag indicates the validity of Country code data in Network Information field.

5.10.3 Format Identifier

The Format Identifier field contains the format identification data of 4 bytes as defined in ISO/IEC 13818-1, whose value is given by SMPTE.

5.10.4 Network Information

Network Information field indicates the type of MPEG2-TS with the identification data defined in DVB specification, as shown below.

Address	Length, bytes	Contents
00 00_{16}	2	Original Network ID
00 01_{16}		
00 02_{16}	2	Transport stream ID
00 03_{16}		
00 04_{16}	2	Service ID
00 05_{16}		
00 06_{16}	3	Country code
00 07_{16}		
00 08_{16}		

Figure 5.17 – Network Information

Original Network ID is a unique identifier for the original network.

Transport stream ID is a unique identifier of a transport stream within an original network.

Service ID field contains a unique identifier of a service within a TS.

The three fields above, *Original Network ID, Transport stream ID, and Service ID* apply to ETSI EN300-468.

The *Country code* field identifies the country which is subject for service. The *country code* value is defined in ISO3166 or ETR162 as code alpha-3.

5.10.5 Format Identifier or Network Information

If both *Format Identifier* and *Network Information* are specified, the controller should use Format Identifier for Identification purposes.

		Format Identifier	Network Information			
			Original Network ID	Transport stream ID	Service ID	Country code
ATSC		O	-	-	-	-
DVB		-	O	*	*	*
ARIB	BS	-	O	*	*	O
	Terrestrial	-	O	*	*	O

O : available, and useful for stream identification.
- : not available.
* : available, but not always required for stream identification.

Figure 5.18 – Format Identifier or Network Information Selector

5.10.6 Stream Format Name

The *Stream Format name* field has 16 bytes. The data starts from the MSB, and uses the ISO-646 character code set. If the data is shorter than 16 bytes, the remainder is filled with FF_{16}. When the stream format name is not valid, the field shall contain all FF_{16}.

5.11 Program Attribute Info Block (8814_{16})

The format of the *Program attribute info block*, which defines information about a broadcast program, is shown in following figure:

Address	Length, bytes	Contents
00 00_{16}	2	compound_length
00 01_{16}		
00 02_{16}	2	info_block_type = (Program_attribute_info_block)
00 03_{16}		
00 04_{16}	2	Primary_fields_length
00 05_{16}		
00 06_{16}	7	Program starting date & time
:		
00 $0C_{16}$		
00 $0D_{16}$	4	Program duration
:		
00 10_{16}		
00 11_{16}	1	parental rating
00 12_{16}	1	program modes
00 13_{16}		Service name
:		
00 xx_{16}		

Figure 5.19 – Program Attribute Info Block

The *primary fields length* describes the total size in bytes of the program starting date and time, program duration, parental rating, program modes and service name fields.

The *Program attribute info block* only specifies attributes of the broadcast program, these values may be inconsistent with AV Content Object attributes. (ex. duration)

If a controller wants to know attributes to control (play, search, etc.) an AV Content Object, the controller should refer to the attribute found in that AV Content Object entry.

5.11.1 Program Starting Date and Time

Program starting date & time field shows the date and time that the program broadcast started.

Address	Length, bytes	Contents
00 00_{16}	2	Year
00 01_{16}		
00 02_{16}	1	Month
00 03_{16}	1	Day
00 04_{16}	1	Hours
00 05_{16}	1	Minutes
00 06_{16}	1	Seconds

Figure 5.20 – Program Starting Date & Time Field

Program starting date & time fields are encoded as BCD.

5.11.2 Program Duration

Program duration field shows the length of the program broadcast in time.

Address	Length, bytes	Contents
00 00_{16}	2	Hours
00 01_{16}		
00 02_{16}	1	Minutes
00 03_{16}	1	Seconds

Figure 5.21 – Program Duration Field

Program duration fields are encoded as BCD.

5.11.3 Parental Rating

The Parental Rating field is defined as follows:

Address Offset	length	msb							lsb
00 00_{16}	1	Rating	Reserved						

Figure 5.22 – Parental Rating Field

When set to 1, the Rating bit specifies that parental rating is applied to this object.

5.11.4 Program Mode

Address Offset	length	msb							lsb
00 00_{16}	1	HD/SD		Replayed		Reserved			

Figure 5.23 – Program Mode Field

HD/SD field shows that the program was broadcast in HD mode or SD mode.

Table 5.9 – HD/SD Field Values

Value	Definition
00_2	SD
01_2	HD
10_2	Reserved for future definition
11_2	No information

Replayed field shows whether the program has been replayed or not.

Table 5.10 – Replayed Field Values

Value	Definition
00_2	Not replayed
01_2	Replayed
10_2	Reserved for future definition
11_2	No information

5.11.5 Service Name

Service name field shows the name of service that broadcast the program.

Table 5.11 – Service Name Field

Offset	Contents
00 00_{16}	Character code type
00 01_{16}	Character code type specific
00 02_{16}	Maximum service name length
00 03_{16}	Service name length
00 04_{16} : :	Service name

The character code type field contains values that are defined in character code info block in AV/C General Enhancement 3.0.

The character code type specific field contains values that are defined in character code info block in AV/C General Enhancement 3.0.

The maximum service name length field specifies the maximum allowable number of characters, not bytes, for the service name field.

The service name length field contains a value that defines the actual number of bytes, not characters, contained in the service name field.

The Service name field contains the actual service name, in a character code format specified by the character code type and Character code type specific fields.

6. Disc Subunit Identifier Descriptor

6.1 Disc Subunit Dependent Information

The following sections define enhancements to the Disc Subunit Identifier descriptor, subunit dependent information, and supported media type specification.

The *Disc_subunit_dependent_information* field of the subunit identifier structure contains specific information that describes the capabilities of the subunit. The subunit dependent information field has the following definition.

6.1.1 Supported Media Type

A new value is defined to specify that the Disc Subunit supports an HDD Device-Type media.

Table 6.1 – Supported media types

Supported_media_type	Meaning
$01xx_{16}$	CD media types
$03xx_{16}$	MD media types
0700_{16}	HDD device type media

6.1.2 Implementation Profile ID

A value of zero indicates that the device supports the AV/C Disc Subunit specification with no further definition of media format, command set, or feature support.

All remaining non-zero values are used to indicate appropriate media type specifications.

Table 6.2 – Implementation profile ID

value	Meaning
00_{16}	Complies with reference document [R7] – no profile specified
All others	Indicates specific profile supported by the subunit.

7. Disc Subunit Status Descriptor

7.1 Plug Configuration Info Block

New video-object, type-specific plug-configuration information is defined in the following sections. This information should be provided using the CONFIGURE command, in the configuration state field.

7.1.1 Video Object Type - Destination Plug Configuration

When a destination plug is configured for recording a Video AV Track, the *object_and_plug_type_specific_information* field in the *plug_configuration_info_block* has the following format:

Address Offset	length	msb							lsb
00 00_{16}	1	Increment_position_number	Reserved						
00 01_{16}	1	Video_signal_mode							
00 02_{16}		video_signal_specific_information							
:									
:									

Figure 7.1 – Video specific destination plug information

The *increment_position_number* bit indicates whether the plug is currently configured to increment the position number when the recording operation is paused. When this bit is set to 1, a new AV Content object entry is created in response to a RECORD(PAUSE) command. This new AV Content object entry is inserted into the associated list immediately after the current AV Content object entry; the position of any entry following the position of this new AV content object entry is automatically incremented by one. When this bit is set to 0, a new AV Content object entry will not be created in response to a RECORD(PAUSE) command.

The *video_signal_mode* and *video_signal_mode_specific_information* fields are defined in section 5.9

NOTE —When a controller writes the bit rate of a stream to be recorded, a disc subunit can use the *video_recording_bit_rate* to estimate the remaining recording time, to reserve loop recording area, or to configure the destination plug to record the stream.

7.1.2 Video Object Type - Source Plug Configuration

When a source plug is configured for playback of a video AV Track, the *object_and_plug_type_specific_information* field has the following format.

Address Offset	length	msb							lsb
00 00_{16}	1	audio_mute	Reserved						

Figure 7.2 – Video specific source plug information

The *audio_mute* bit indicates whether muting of the audio output signal is in effect (=1) or not (=0). The mute function is applied to the audio signal in the case of some trick play operations (fast forward, fast reverse). If the subunit does not support this feature, then this bit shall be set to 0 (the audio signal cannot be muted).

Video object type - source plugs are not configured in the same manner as video object type - destination plugs. Video object type- source plug configuration is determined by the current plug - object association.

8. Disc Subunit Commands

8.1 Create AV Track ($D5_{16}$)

The CREATE AV TRACK control command is used to create both a new AV track and a new AV Content Object Descriptor. It is useful to support the Record command, loop subfunction.

Before the CREATE AV TRACK command is issued, it is necessary to configure the destination plug using the CONFIGURE command. This specifies the AV Content object type and other configuration information in the plug configuration info block.

After the destination plug is configured, the destination plug must be associated with the specified list using the SET PLUG ASSOCIATION command when the plug is not already associated with a list. This association is not automatic.

The plug configuration and list-plug association are accomplished before the CREATE AV TRACK command is issued.

The control command has the following frame:

	length	msb							lsb
opcode	1	CREATE AV TRACK ($D5_{16}$)							
operand[0]	1	Result							
operand[1]	1	Subfunction_1							
operand[2]	1	Reserved							
operand[3]	2	destination_plug							
operand[4]									
operand[5]		new_object_entry_identifier							
:									
:									
:		AV_track_size							
:									
:									

Figure 8.1 – Create AV track control command format

	msb							lsb	
opcode	CREATE AV TRACK ($D5_{16}$)								*Common*
operand[0]	Result								*Command*
operand[1]	Subfunction_1								*header*
operand[2]	Reserved								*part*
operand[3]	destination_plug								*plug_*
operand[4]									*Identifier_part*
operand[5]	new_object_entry_identifier								*original*
:									
:									
:	AV_track_size								*original*
:									
:									

Figure 8.2 – Create AV track control command format

Subfunction_1 specifies the format of the returned object identifier. The currently defined values for subfunction_1 are as follows. Notice that these values are identical to the descriptor_type values from reference document [R4] section 10.1. It is possible to easily extend this table to support any future descriptor types that may be specified in the AVC general model and its enhancements.

Table 8.1 – Subfunction_1 values

value	Meaning
20_{16}	Return listID:object_position.
21_{16}	Return rootID:root type:object_ID
All others	Reserved for future speciciation

The destination_plug field specifies the appropriately configured subunit plug for this operation.

The *new_object_entry_identifier* field returns the object identifier of the newly created AV Content Object descriptor, an object entry in the plug-associated list. This field will have the same format as the descriptor_type specific information specified in reference document [R4] section 10.1.3. That is either list_ID:object_position or root_list_ID:list_type:object_ID.

The *AV_track_size* field contains the *size_indicator_type* and *size_indicator_type_specific* fields of a *size_indicator_info_block*.

NOTE —Because the returned new_object_entry_indentifier field can be a variable size, it is important that the controller create a correct command frame of proper size. The controller must also anticipate a response frame of proper size to accommodate the expected return information and place the AV_track_size field at the correct address offset.

The result field in the response may have one of the following values:

Table 8.2 – Result field in the response

Response frame type	result	result code name	meaning
ACCEPTED	00_{16}	Success	Successful completion
	All other values		Reserved for future specification
REJECTED	FF_{16}	Unknown	An unknown error occurred
	All other values		Reserved for future specification

8.2 Erase Command

This section defines a proposed enhancement to the ERASE control command. Subfunction_1 field has been updated.

	length	msb							lsb
opcode	1	ERASE (40_{16})							
operand[0]	1	Result							
operand[1]	1	Subfunction_1							
operand[2]	1	Reserved							
operand[3]		Erase_type_specific							
:									
:									

Figure 8.3 – Erase control command

8.3 Record (loop)

The section specifies loop recording rec_mode of RECORD control command. This is different than the Time Machine rec_state mentioned in AV/C Disc Subunit Model and Command Set, section 10.18. Loop recording does not buffer data until a record command is issued. In addition, loop recording actually records to an AV Track, which allows the controller to loop record very large amounts of AV content.

The controller may specify a particular AV Track to use during record operation. The controller may use the SET PLUG ASSOCIATION command to specify which object list shall be used during the record operation.

When the record position advances to the end of track, the record position is not wrapped around to the start of track. Instead, old data is shifted out of the track and new data is appended at the end of track.

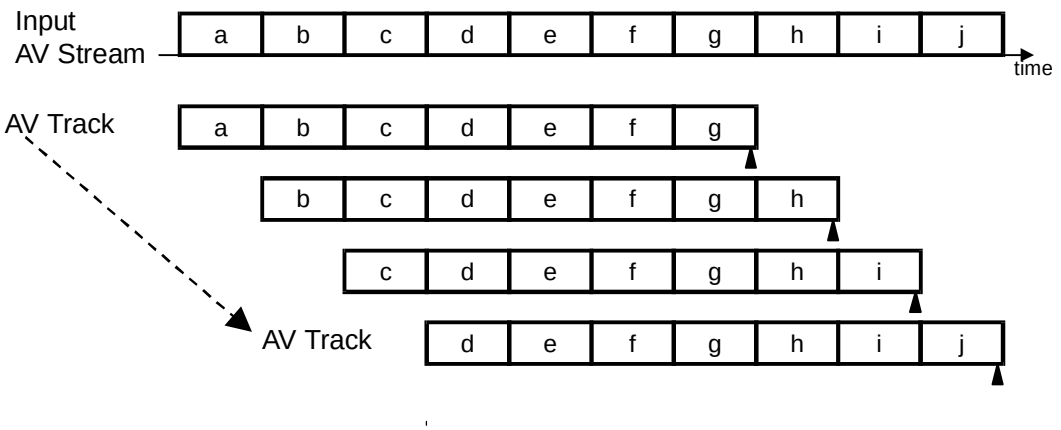

Figure 8.4 – Record loop mode

Recording continues until stopped by the controller. In most record functions the size of the AV Track increases as more AV content is recorded. In the loop record function the AV Track size does not increase.

The subfunction_2 field specifies the rec_mode, indicating this variation of the RECORD command. The new value defined for this field is listed below:

Table 8.3 – Rec_mode values

rec_mode	Value	Meaning
loop	06_{16}	Record new data in the AV track using loop record mode. Any existing data may be overwritten by the new data. The AV track size does not change.

The new_object_position_number field shall be set to all FF_{16} bytes when rec_mode=loop. No new object is created by loop record mode operation and so this field shall not be updated as part of the response frame.

The *start_position* field is an edit_location_indicator_part that defines the beginning of the loop record operation. When *start_position* contains a value of all FF16 bytes start_position does not specify a location. In this case, recording should begin from the previous record position as indicated by the recording position info block (defined in section 5.3) placed in AV Content Object at the end of the last record operation.

Operand offset	length	msb							lsb
00_{16}		start_position							
:									
:									

Figure 8.5 – Rec_mode = loop

The control_position indicator part specifies the start_position of the AV Track where recording shall begin. The record opreation begins at the start_position and the record position advances until recording is complete.

The result field in the response may have one of the following values:

Table 8.4 – Result field in the response

Response frame type	result	result code name	meaning
ACCEPTED	00_{16}	Success	Successful completion
	All other values		Reserved for future specification
REJECTED	FF_{16}	Unknown	An unknown error occurred
	All other values		Reserved for future specification

8.4 Set Plug Association

The ASSOCIATE LIST WITH PLUG command is now renamed to SET PLUG ASSOCIATION. This name change was made because the previous name did not reflect the possibility to associate a specific object with a plug as well as associate a list with a plug.

This page is left blank intentionally

Annexes

Annex A: Informative Annex

As discussed at the Las Vegas meeting, the purpose of this enhancement document is to support the first HDD Device Type Profile scheduled for completion by January 2000. In order to complete this proposal within the timeframe of the HDD Device Type Profile, several incomplete sections of this proposal have been removed from the main document and placed here in this annex.

It must be stressed that this annex is NOT NORMATIVE. It contains work that is not yet completed and, it should not be used for product specification.

The sections contained here are merely a record of the work to date on various topics that are judged by the Work Project Leader as being significantly incomplete and non-essential to the completion of the HDD Device Type profile.

A.1 Positioning within an AV Track

Many record and editing operations rely on positioning within the AV Track. The result of a positioning command operation depends on implementation. For example, if an AV track contains MPEG encoded data, when a command specifies a location such as a particular AV frame in this AV track, the disc subunit may or may not position at the nearest GOP boundary due to MPEG encoding format.

Position within an AV Track depends on the encoding format of the data being edited and the specific device implementation. The device may support editing position for AV segment or AV track. The device may also support byte positioning for editing operation. Any combination of editing positions may be supported as an implementation choice.

A.2 Positioning in Loop Recording Mode.

Typically the position in an AV track is specified as a track relative position and is measured from the start of the AV track. However, this causes problems when a track relative position is used in loop recording mode. As new AV data is recorded, the track relative position of the previously recorded AV data is moving backward in the direction of the start of the AV Track. So it is difficult to specify the position of AV data within the AV Track.

Track relative positioning can be used for the playback and recording positions in loop recording mode. Assuming an equal rate for both playback and record, as new AV data is input and old AV data is deleted, the record and playback position values within the AV track do not change. The only change in positioning is relative to the AV stream content over time.

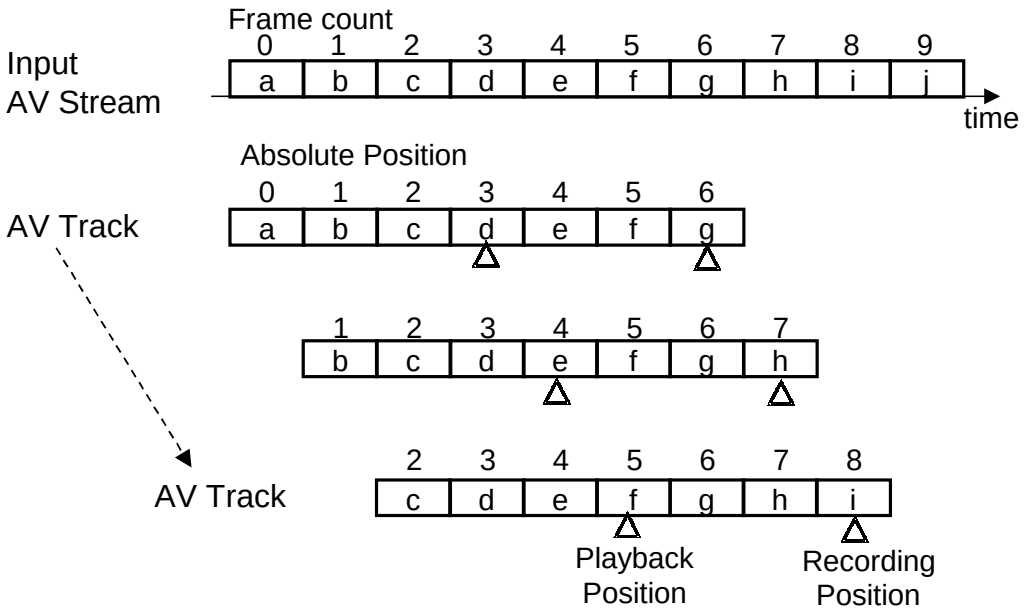

Figure A.1 – Positioning in record loop mode

NOTE —A new positioning mode is required to identify a specific AV Frame or AV Segment within the AV Track during loop recording mode operation.

A.3 Record (overwrite_new)

In the "record overwrite_new" function, new AV content is recorded on an AV track, beginning at the start_position specified by the RECORD command. The record position begins at the start_position and advances until recording is complete. When the record position advances to the end of the track, recording continues and the track size increases to contain the newly recorded AV content. Recording may continue until available media is exhausted.

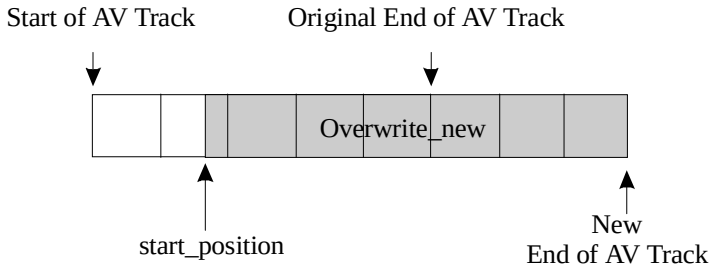

Figure A.2 – Record overwrite_new

The "record overwrite_new" command operation does not modify other existing AV tracks.

The "record append" function is very similar to the "record overwrite_new" function with one key difference: the record overwrite_new function modifies existing data, whereas the record append operation does not. In record append operation, new data is recorded at the end of the track. There is no new

subfunction value for “record append.” Instead, the operation is specified as a record overwrite_new command with the start_position equal to the end of track position.

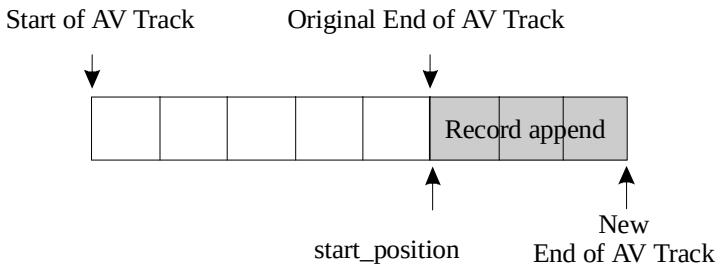

Figure A.3 – Record append

A.4 Record (overwrite_track)

In “record overwrite_track,” new data is recorded at the start_position and continues until the end of the track. The “overwrite_track” function is very similar to the record “overwrite_new” function. The key difference is that the overwrite_track operation does *not* change the size of the track being recorded.

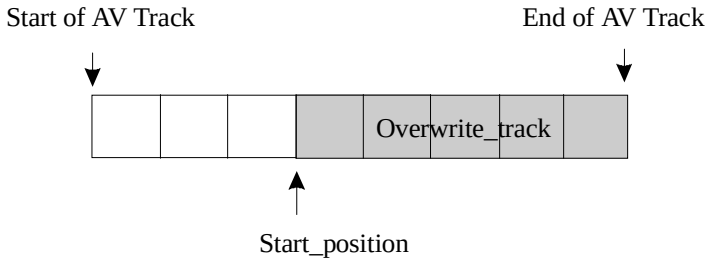

Figure A.4 – Record overwrite_track

A.5 Record (overwrite_segment)

The subfunction “record overwrite_segment” overwrites data previously recorded on the track, starting from the in-point and ending at the out-point. The RECORD command uses an edit_range_specification part to specify the in-point and out-point of the AV segment to be recorded.

Unlike most record functions where the size of the AV track increases as more AV content is recorded, the record “overwrite_segment” function does not change AV track size. The figure below illustrates the record overwrite_segment operation.

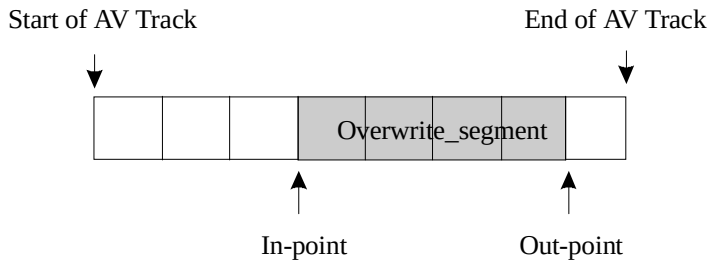

Figure A.5 – Record overwrite_segment

A.6 Record (insert)

Additional AV content can be inserted into an existing AV Track using the function "record insert." A fundamental characteristic of the record insert function is the change in positioning of existing data in relation to the start of track when new data is inserted into the AV track.

The RECORD command specifies the start_position of the AV Track to be recorded. The record position begins at the start_position and advances until recording is complete. All existing AV content from the start_position to the end of track remains unchanged as new data is inserted ahead of it. The track size increases to contain the newly recorded AV content. Recording may continue until available media is exhausted. The following figure illustrates the insert function using labeled, generic AV data as an example.

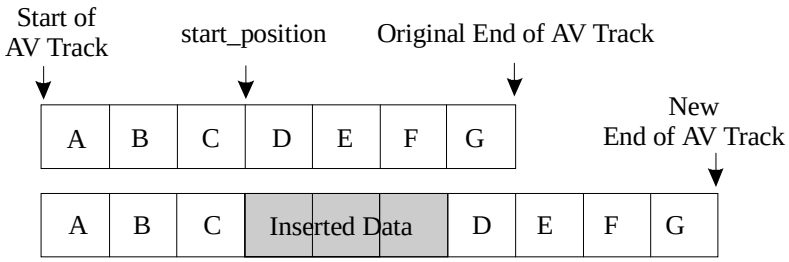

Figure A.6 – Record insert

A.7 Record Frame

This proposal defines a new command called RECORD FRAME. The RECORD FRAME command is useful for editing devices that can access encoding-specific frame boundaries in the recorded AV Track data as specified by the plug configuration.

Valid defined rec_mode values for RECORD FRAME command are *new*, *overwrite_new*, and *insert.*

The RECORD FRAME command uses the HMSF position indicator type = 00_{16} exclusively.

There is no status or notification ctype for the RECORD FAME command

Data type and format must be specified in plug configuration.

A.8 Divide Segment

The DIVIDE SEGMENT command splits the specified AV segment into two separate AV segments at the position specified by the command. For example, AV segment[1] can be divided into AV segment[1] and AV segment[2]. All AV segments from the original AV segment[2] and beyond increment their segment numbers by one. The DIVIDE SEGMENT operation is illustrated in the figure below.

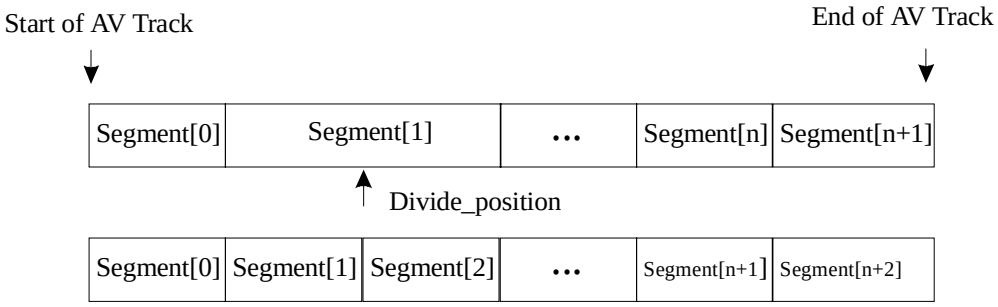

Figure A.7 – Divide segment

A.9 Combine Segment

The COMBINE SEGMENT command merges two adjacent AV segments into a single AV segment. The command operands specify the first of two AV segments to be combined. For example, if AV segment[1] is specified, then AV segment[1] and AV segment[2] are combined to become AV segment[1]. All AV segments beyond the original AV segment[2] decrement their segment numbers by one. The COMBINE SEGMENT command operation is illustrated in the figure below.

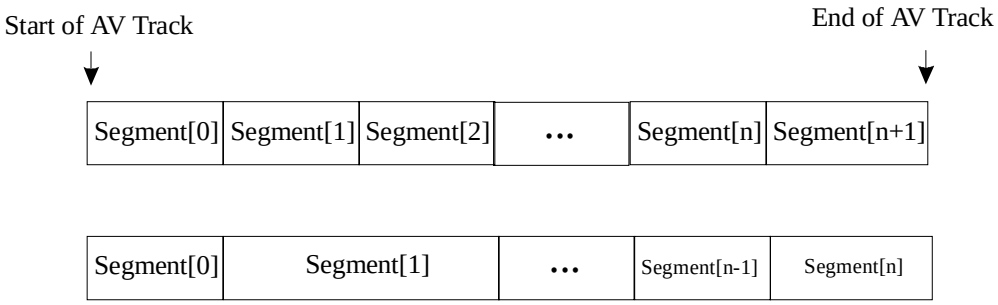

Figure A.8 – Combine segment

A.10 Combine Segments

The COMBINE SEGMENTS command joins two adjacent AV Segments into a single segment as described in section A.9. The COMBINE SEGMENTS operation removes the segment marker entry that specifies the boundary between the two adjacent segments as specified by the command operands. The COMBINE SEGMENTS command is illustrated in the following figure.

	length	msb							lsb
opcode	1	COMBINE SEGMENTS (XX_{16})							
operand[0]	1	Result							
operand[1]	1	Reserved							
operand[2]	1	Reserved							
operand[3]		descriptor_identifier							
:									
:									
:		segment_number							
:									
:									

Figure A.9 – Combine segments

The descriptor_identifier field specifies the object that contains the segment table to be modified and describes the track containing the segments to be combined.

The segment_number field is a relative segment count indicator type and specifies the segments to be combined. These segments must be adjacent according to their segment marker entries in the segment marker info block. So segment[n] and segment[n+1] are combined to become the new segment[n].

A.11 Divide Segment

Divide segment divides an existing AV segment at the point specified by the divide_position field. The result is two neighboring AV segments, as described in section A.8. The Divide Segment command creates a new segment marker that defines the boundary between the two newly created segments. The Divide Segments command is illustrated below.

	length	msb							lsb
opcode	1	DIVIDE SEGMENT (XX_{16})							
operand[0]	1	Result							
operand[1]	1	Reserved							
operand[2]	1	Reserved							
operand[3]		descriptor_identifier							
:									
:									
:		segment_number							
:									
:									
:		Divide_position							
:									
:									

Figure A.10 – Divide segment

The descriptor_identifier field specifies the object that contains the segment table to be modified and describes the track that contains the segment to be divided.

The segment_number field is a relative segment count indicator type and specifies the segment to be divided.

The divide position field is an *edit_position_indicator* that specifies the location where a new segment marker will point. The new segment marker divides the implied segment at the edit position. The *edit_position_indicator* is a new type for segment relative offset addressing. The edit position is measured relative to the start of the segment.

Table A.1 – New indicator_type

Indicator type	Meaning
00_{16}	Relative_HMSF_count
01_{16}	Relative_segment_count
02_{16}	Relative_byte_count
03_{16}	Segment_relative_offset
04_{16} - FF_{16}	reserved for future specification

The segment relative offset indicator contains the following information.

	length	msb							lsb
:	1	Indicator_type = 03_{16}							
:	1	+/-	Hours						
:	1	Minutes							
:	1	Seconds							
:	1	frames							

Figure A.11 – Segment_relative_offset Indicator type

If the specified divide_position is not within the specified segment_number, then the subunit will respond with a REJECTED response code.

A.12 Record

The section specifies the details of the new recording modes described in section 5. The subfunction_2 field specifies the rec_mode, indicating which variation of the RECORD command is to be executed. The newly defined values for this field are listed below:

Table A.2 – subfunction_2 Rec_mode values

rec_mode	Value	Meaning
overwrite_new	02_{16}	Records new AV data, beginning at the specified start_position. Existing data is overwritten by the new data. Append operation can be achieved by specifying the end of track as the start_position.
overwrite_track	03_{16}	Records new data in the AV track, beginning at the position indicated by the start_position field. Existing data is overwritten by the new data. The AV track size does not change.
overwrite_segment	04_{16}	Records new data in the AV segment specified by the edit_range_specification part. Existing data is overwritten by the new data. The AV track size does not change.
insert	05_{16}	Inserts new AV data, beginning at the specified position. No existing data is modified except its position from the start of track

The new sub_function_2 values are explained in more detail below:

A.12.1 Rec_mode = overwrite_new

Address	Length, bytes	Contents
00 00_{16}		start_position
:		
:		

Figure A.12 – Rec_mode = overwrite_new

The *start_position* field is a edit_location_indicator_part that defines the beginning of the overwrite operation.

A.12.2 Rec_mode = overwrite_track

Address	Length, bytes	Contents
00 00_{16}		start_position
:		
:		

Figure A.13 – Rec_mode = overwrite_track

The *start_position* field is a edit_location_indicator_part that defines where the beginning of the overwrite operation. Recording can continue until the end of track. The size of the track is not changed by overwrite_track

A.12.3 Rec_mode = overwrite_segment

The *in-point* and *out-point* fields are a edit_range_specification_part that defines the location of the overwrite operation.

Address	Length, bytes	Contents
00 00_{16}		in-point
:		
:		
:		out-point
:		
:		

Figure A.14 – Rec_mode = overwrite_segment

The *in-point* field is a edit_location_indicator_part that defines the beginning of the overwrite operation. The *out-point* field is a edit_location_indicator_part that defines the end of the overwrite operation.

A.12.4 Rec_mode = insert

Address	Length, bytes	Contents
00 00_{16}		start_position
:		
:		

Figure A.15 – Rec_mode = insert

The *start_position* field is a edit_location_indicator_part that defines the beginning of the insert operation.

NOTE —The three overwrite rec_modes for new, track and segment may be combined or remain separate subfunctions pending further discussion.

A.13 Record Frame

This proposal defines a new command called RECORD FRAME. The record frame command is useful for editing devices that have knowledge of and access to encoding specific frame boundaries in the recorded AV Track data as specified by the plug configuration.

The RECORD FRAME command has the following format.

	length	msb							lsb
opcode	1	RECORD FRAME (XX_{16})							
operand[0]	1	result							
operand[1]	1	reserved							
operand[2]	1	subfunction_2							
operand[3]	2	destination_plug							
operand[4]									
operand[5]		new_object_position_number							
:									
:									
:		rec_mode_specification							
:									
:									
:		number_of_frames							
:									

Figure A.16 – Record frame command

	msb							lsb	
opcode	RECORD FRAME (XX_{16})								*common command header part*
operand[0]	result								
operand[1]	reserved								
operand[2]	subfunction_2								
operand[3]	destination_plug								*plug_ identifier_part*
operand[4]									
operand[5]	new_object_position_number								*original*
:									
:									
:	rec_mode_specification								*control_ position_ indicator_part*
:									
:									
:	number_of_frames								*Opcode Specific part*
:									
:									

Figure A.17 – Record frame command

The *subfunction_2* operand specifies the *rec_mode*, which has the same format and meaning as the RECORD control command.

The destination_plug and *new_object_position_number* operands have the same format and meaning as the RECORD control command.

The *rec_mode_specification* is a edit_location_indicator_part. Its format depends on the value of rec_mode in the subfuncion_2 field.

The *number_of_frames* operand specifies the number of frames to be recorded as shown below.

Operand offset	length	msb							lsb
00_{16}	1	length of number_of_frames							
01_{16}		number_of_frames							
:									
:									

Figure A.18 – Number of frames operand

operand offset	msb							lsb	
00_{16}	length of number_of_frames								*Opcode Specific part*
01_{16}	number_of_frames								
:									
:									

Figure A.19 – Number of frames operand

The *length_of_number_of_frames* field specifies the length of the next field as a number of bytes.

The *number_of_frames* field specifies the number of frames to be recorded in the AV Track at the specified start_position.