TA Document 1999005

AV/C Bulletin Board Subunit General Specification Version 1.0

August 4, 1999

Sponsored by:
1394 Trade Association

Approved for Release by:
This document has been approved for release by the 1394 Trade Association Board of Directors.

Abstract:
The AV/C Bulletin Board Subunit provides information that can be shared with other devices linked on a 1394 network. An AV/C Bulletin Board Subunit contains the information about the device that the Bulletin Board subunit belongs to. The information is represented by the descriptor mechanism which is defined in the AV/C Digital Interface Command Set General Specification version 3.0 and Enhancement to the AV/C General Specification 3.0, version 1.0.

Keywords:
AV/C General Specification, Descriptor, AV/C Bulletin Board Type Specification.

Copyright © 1996-1999 by the 1394 Trade Association.
Regency Plaza Suite 350, 2350 Mission College Blvd., Santa Clara, CA 95054, USA
http://www.1394TA.org
All rights reserved.

Permission is granted to members of the 1394 Trade Association to reproduce this document for their own use or the use of other 1394 Trade Association members only, provided this notice is included. All other rights reserved. Duplication for sale, or for commercial or for-profit use is strictly prohibited without the prior written consent of the 1394 Trade Association.

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 approved 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.

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

Table of Contents

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

List of Figures

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

List of Tables

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

1. Preface

1.1 Purpose and Scope

This document is the specification for the AV/C Bulletin Board Subunit. The purpose and scope of this feature are summarized below.

Purpose: The purpose of AV/C Bulletin Board Subunit is to provide information that can be shared with other devices on a 1394 network. The information must be related to the unit where the subunit resides.

Scope: This document defines the Bulletin Board Subunit model, specific data structures and command sets. The AV/C Bulletin Board Subunit uses the descriptor mechanism defined in the AV/C Digital Interface Command Set General Specification version 3.0 and the Enhancements to AV/C General Specification 3.0, version 1.0.

There are separate documents that define board-type specific data structures.

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

2. References

2.1 Contact Information

Much of the information in this document is preliminary and subject to change. Members of the AVWG are encouraged to review and provide inputs for this proposal. For document status updates, please contact:

Mari Horiguchi Workitem Project Leader
i.LINK Development Department SUPC Sony Corporation
Shinagawa Tec. Shinagawa INTERCITY C Tower W/9F
2-15-3, Kounan, Minato-ku, Tokyo
108-6201 Japan
E-Mail: marik@arch.sony.co.jp
Phone: +81-3-5769-5336
Fax: +81-3-5769-5834

For technical comments, please contact:

Technical contributor
Jon Brelin
Sony, America
3300 Zanker Rd
San Jose, CA 95124
E-Mail: jon.brelin@am.sony.com
Phone: 408-955-6731
Fax: 408-955-6236

Other technical officer
Hisato Shima
Sony, America
3300 Zanker Rd.
San Jose, CA 95124
E-Mail: hisato.shima@am.sony.com
Phone: 408-955-5524
Fax: 408-955-6236

You can also submit comments using the 1394 TA reflector. Its web site address is:

www.1394ta.org

Go to the members-only section and select the AVWG reflector. You will need to have signed up for the reflector.

The documents referenced herein may be obtained from the following organizations:

2.1.1 1394 Trade Association (1394 TA)

The 1394 Trade Association can be contacted via the references provided on the cover page of this and all AV/C specification documents.

2.2 1394 Trade Association Specifications

[R1] AV/C Digital Interface Command Set General Specification version 3.0

[R2] Enhancements to the AV/C General Specification 3.0, version 1.0.

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

3. Change History

The following table shows all changes made to this document.

Date	Version	Changes	Editor(s)
July 1, 1999	1.0	Original Version	Mari Horiguchi, Jon Brelin

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

4. Definitions and Abbreviations

4.1 Conformance glossary

Several keywords are used to differentiate between different levels of requirements and optionality, as follows:

4.1.1 expected: A keyword 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.

4.1.2 may: A keyword that indicates flexibility of choice with no implied preference.

4.1.3 shall: A keyword indicating a mandatory requirement. Designers are required to implement all such mandatory requirements to ensure interoperability with other products conforming to this specification.

4.1.4 should: A keyword indicating flexibility of choice with a strongly preferred alternative. Equivalent to the phrase “is recommended.”

4.2 Technical glossary

4.2.1 AV/C Controller: A device that issues AV/C commands to other devices.

4.2.2 AV/C Target: A device that receives AV/C commands.

4.2.3 board: A group of object lists and their associated object entries that are controlled by the Bulletin Board Subunit. The information therein is shared between devices.

4.2.4 Board Entry Descriptor: An object entry descriptor that is used to identify a bulletin board of a particular type. A Board Entry Descriptor has a list ID of an Information List Descriptor.

4.2.5 Board List Descriptor: An object list descriptor that contains Board Entry Descriptors which identify Bulletin Boards of a particular type.

4.2.6 board type: A type of Bulletin Board that is defined by a separate board type specification.

4.2.7 descriptor: The general term for a data structure that describes something, such as the subunit, individual pieces of content on media, etc.

4.2.8 device: The physical instantiation of a consumer electronic device, e.g., a camcorder or a VCR, within a Serial Bus node.

4.2.9 GUID: Global Unique ID of the device. This is the EUI-64 value as described in the 1394-1995 specification, and in the AV/C Digital Interface Command Set General Specification version 3.0.

4.2.10 info_block: Information blocks are an enhancement to the descriptor model. Each information block contains a collection of related data fields; info blocks can also contain other info blocks.

4.2.11 Information Entry Descriptor: An object entry descriptor that contains a parcel of information that makes up a Bulletin Board. Information Entry Descriptors are contained in Information List Descriptors.

1394 TRADE ASSOCIATION THE MULTIMEDIA CONNECTION
Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

4.2.12 Information List Descriptor: An object list descriptor that contains Information Entry Descriptors that make up a Bulletin Board. Its list ID is within the Board Entry Descriptor as a child list ID, or in the subunit identifier descriptor as a root list ID.

4.2.13 posting device: An AV/C controller that posts its information to the Bulletin Board Subunit.

4.2.14 reading Device: An AV/C controller that reads information from the Bulletin Board Subunit.

4.2.15 Subunit Identifier Descriptor: A data structure that contains attribute information about the subunit that it refers to; the descriptor content and format can vary based on the type of subunit that it describes.

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

5. The Bulletin Board Subunit Model

5.1 Function

The AV/C Bulletin Board Subunit supports the sharing of information with other devices on a 1394 network. The shared information is about the device that the Bulletin Board subunit belongs to. There is only one Bulletin Board Subunit in a unit.

The AV/C Bulletin Board Subunit provides information using the descriptor mechanism that is defined in the AV/C Digital Interface Command Set General Specification version 3.0 [R1] and Enhancements to the AV/C General Specification 3.0, version 1.0 [R2].

The contents of the Bulletin Board should survive bus resets and power-off.

5.2 Bulletin Board Descriptors

5.2.1 Unique descriptor naming

In the AV/C Digital Interface Command Set General Specification version 3.0 [R1], the Object List Descriptor and Object Entry Descriptor data structures are defined. Though using the same data structures, this specification uses Bulletin Board specific descriptor names. This document terms *Board List Descriptors* and *Information List Descriptors* as Object List Descriptors, and *Board Entry Descriptors* and *Information Entry Descriptors* as Object Entry Descriptors.

Board List Descriptors contain only Board Entry Descriptors, and Information List Descriptors contain only Information Entry Descriptors. This document also refers to a Bulletin Board specific Subunit Identifier Descriptor. These descriptors are combined in the Bulletin Board subunit to produce the data structures necessary to share information between devices.

The following five clauses describe the descriptors that are used to make up the bulletin board.

5.2.2 Information Entry Descriptor

A piece of information and its ID are contained in an Information Entry Descriptor. This descriptor holds the primary information that the Bulletin Board Subunit was designed for.

5.2.3 Information List Descriptor

Information List Descriptors contain Information Entry Descriptors and define attributes common to all entries. Each information list is considered to be a board of a specific type. All entries within the list are considered to be of one type. An Information List Descriptor allows for data that describes the list specifically, such as whether the list is Read Only or Write Enabled, or other information that differentiates this list from others of the same type.

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

5.2.4 Board Entry Descriptor

A Board Entry Descriptor contains list IDs of an Information List Descriptor. Board Entry Descriptors are entries in a Board List Descriptor and allow Board List Descriptors to specify multiple Boards within the list.

5.2.5 Board List Descriptor

A Board List Descriptor contains one or more Board Entry Descriptors and is used to contain a list of board entries of the same type. Board List Descriptors and their associated Board Entry Descriptors are used for Bulletin Boards where there is more than one board per board type.

5.2.6 Subunit Identifier Descriptor

The Bulletin Board Subunit contains a Subunit Identifier Descriptor that contains root list ID information that points to bulletin board list structures. The Subunit Identifier Descriptor is fully defined in the AV/C Digital Interface Command Set General Specification version 3.0 [R1].

5.3 Structure Model

Logically, the Bulletin Board Subunit consists of one or more types of boards. Each board type has one or more boards associated with that type. Each board type has a specific type of information that it shares. The type of information is defined in each Board Type Specification.

5.3.1 Board types

A device can embody all or a subset of defined board types in its own Bulletin Board Subunit. Board types are identified by an eight-bit value. Supported board types are defined in separate board type specifications. The following table shows board types that presently exist. The values in this table apply to various fields in this document.

Table 5.1 – Existing board types

Value	Board type
00_{16}	Reserved
01_{16}	Resource Schedule Board
02_{16} – FF_{16}	Reserved for future specification

5.3.2 List types

Information lists and Board lists can be defined as Read Only or Write Enabled. The device that contains the Bulletin Board Subunit can write to a read only list.

A Read Only list can only be read by a device external to the device that contains the Bulletin Board Subunit. Since the device that contains the Bulletin Board Subunit maintains these lists, *only it can write to a read only list*. This is the only exception to Read Only list types. Any external device can write to a Write Enabled list. Note that the AV/C Digital Interface Command Set General Specification version 3.0 [R1]

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

specifies that only one device can write to a list at one time while many devices can read from a list at one time.

The following table shows values for supported list types.

Table 5.2 – List Descriptor list_type value assignments

Value	List type name	Comments
00_{16} – $7F_{16}$	--	Reserved for general definitions
80_{16}	Read Only Board List	Cannot be written by external controller
81_{16}	Write Enabled Board List	Can be written by internal or external controller
82_{16}	Read Only Info List	Cannot be written by external controller
83_{16}	Write Enabled Info List	Can be written by internal or external controller
84_{16} – FF_{16}	--	Reserved

5.3.3 Entry types

There is only one entry type for a Board List Descriptor – a *Bulletin Board*, and there is only one entry type for an Information List Descriptor – *Information*. The following table specifies the value for these entry types:

Table 5.3 – Board/Information List Descriptor entry types

Value	Entry type
00_{16} – $7F_{16}$	Reserved for general definitions
80_{16}	Bulletin Board
81_{16}	Information
82_{16} – FF_{16}	Reserved

5.3.4 List IDs

Each list descriptor has a unique list ID. In the Subunit Identifier Descriptor, a root list ID is used to identify an Information List Descriptor for structures that have one board per board type, and a Board List Descriptor for structures that have multiple boards per board type. For Information List Descriptors pointed to by Board Entry Descriptors, a child list ID is used. Child list IDs are also used in Information Entry Descriptors to point to hierarchical data structures. Root list IDs and child list IDs have a different range of values. These values are shown in the following table.

1394 TRADE ASSOCIATION THE MULTIMEDIA CONNECTION
Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

Table 5.4 – List ID value assignments

Range of values	List definition
0000_{16} -1000_{16}	Reserved in AV/C Digital Interface Command Set General Specification version 3.0
1001_{16} -$10FF_{16}$	Root list ID, assigned for each board type
1100_{16} -$1FFF_{16}$	Reserved
2000_{16} -$3FFF_{16}$	Child list ID, assigned by the Bulletin Board Subunit
4000_{16} -$FFFF_{16}$	Reserved in AV/C Digital Interface Command Set General Specification version 3.0

This document defines the Bulletin Board specific fields in the Subunit Identifier Descriptor, the Board List Descriptor, Board Entry Descriptor, Information List Descriptor, and Information Entry Descriptor. The Bulletin Board specific fields have some board type specific fields, which are defined in separate Board Type Specifications.

5.3.5 Object IDs

Object IDs are specified based on the GUID of the posting device and a record ID assigned by that device. Only Information Entry Descriptors have Object IDs. Board Entry Descriptors do not contain Object IDs.

5.3.6 Structure Combinations

The Bulletin Board Descriptors can be structured to support one board per board type, or multiple boards per board type. The structure is based on the settings of the attribute fields in the descriptors. Two structure models are presented in the following clauses to represent these types. These structures can also be expanded to any level suitable to the specific type of Bulletin Board's data structuring requirements. The choice of structure type is dependent on each Bulletin Board Type specification.

5.3.6.1 Structure model for a board type that has only one board

If a bulletin board type contains only one bulletin board of that type, a two-tier structure is used. This structure is shown below:

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

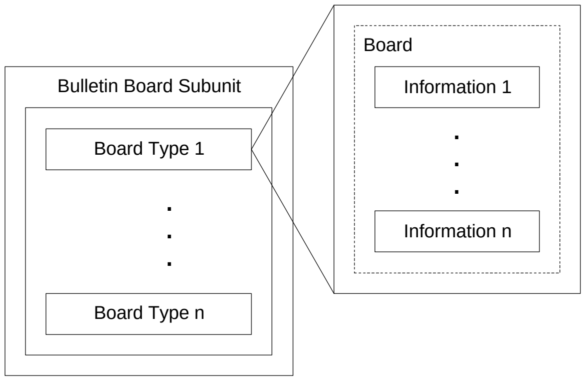

Figure 5.1 – One Bulletin Board per board type

More specifically, the following figure shows a summary of the descriptors, and the values necessary within those descriptors to create this kind structure.

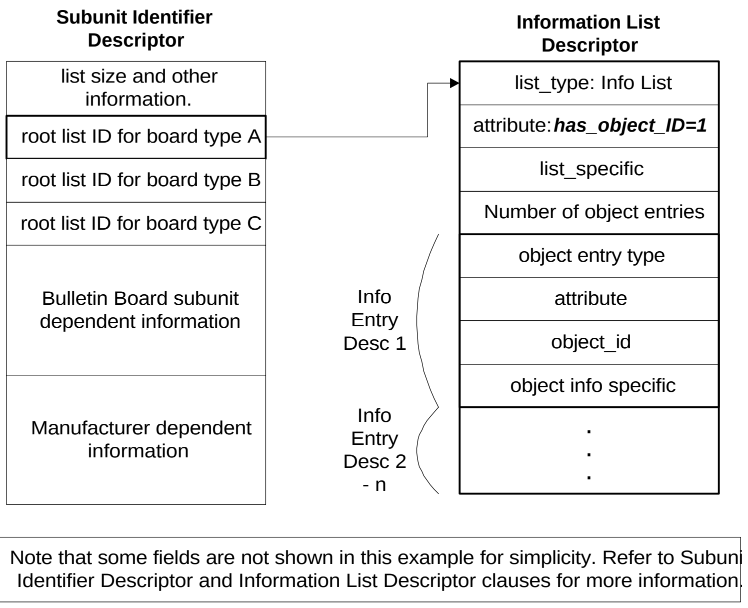
Subunit Identifier Descriptor
list size and other information.
root list ID for board type A
root list ID for board type B
root list ID for board type C
Bulletin Board subunit dependent information
Manufacturer dependent information
Information List Descriptor
list_type: Info List
attribute: has_object_ID=1
list_specific
Number of object entries
object entry type
attribute
object_id
object info specific
Info Entry Desc 1
Info Entry Desc 2 - n
Note that some fields are not shown in this example for simplicity. Refer to Subuni
Identifier Descriptor and Information List Descriptor clauses for more information.

Figure 5.2 – Structure model for the board type that has only one board

Note that there are no Board List Descriptors or Board Entry Descriptors in this configuration. In this case, the Subunit Identifier Descriptor, which contains root list IDs for board types, refers to only one board.

5.3.6.2 Structure model for a board type that has one or more boards

The Bulletin Board subunit can also support multiple boards of a particular type. The structure for multiple boards per board type is depicted in the following figure.

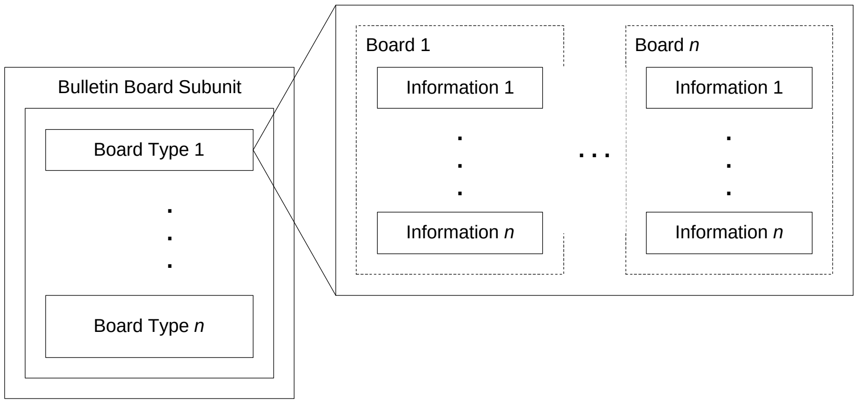

Figure 5.3 – Multiple Bulletin Boards per board type

More specifically, the following figure shows a summary of the descriptors, in a three-tier layout and the attribute values necessary within those descriptors to create this kind structure.

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

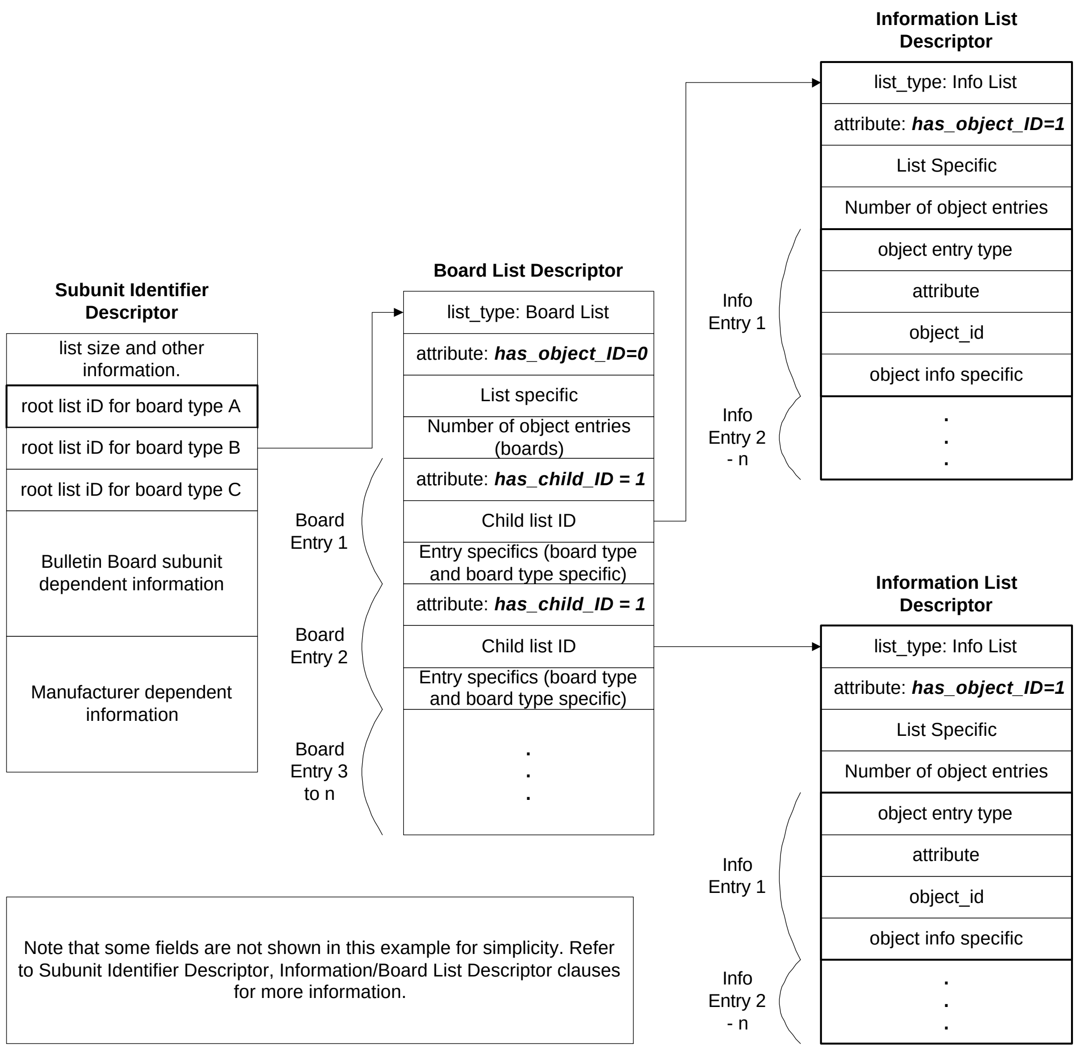
Subunit Identifier Descriptor
list size and other information.
root list iD for board type A
root list iD for board type B
root list iD for board type C
Bulletin Board subunit dependent information
Manufacturer dependent information
Board List Descriptor
list_type: Board List
attribute: has_object_ID=0
List specific
Number of object entries (boards)
attribute: has_child_ID = 1
Child list ID
Entry specifics (board type and board type specific)
attribute: has_child_ID = 1
Child list ID
Entry specifics (board type and board type specific)
Board Entry 1
Board Entry 2
Board Entry 3 to n
Information List Descriptor
list_type: Info List
attribute: has_object_ID=1
List Specific
Number of object entries
object entry type
attribute
object_id
object info specific
Info Entry 1
Info Entry 2 - n
Information List Descriptor
list_type: Info List
attribute: has_object_ID=1
List Specific
Number of object entries
object entry type
attribute
object_id
object info specific
Info Entry 1
Info Entry 2 - n
Note that some fields are not shown in this example for simplicity. Refer to Subunit Identifier Descriptor, Information/Board List Descriptor clauses for more information.

Figure 5.4 – Structure model for the board type that has one or more boards

The Board List Descriptor is "placed" between the Subunit Identifier Descriptor and Information List Descriptors. In this case, the Subunit Identifier Descriptor contains root list IDs for a Board List Descriptor, which refers to one or more boards.

The attribute settings in the Information List Descriptors and Information Entry Descriptors in this structure are the same as presented in section 5.3.6.1.

5.4 Control Model

5.4.1 Controllers

In the Bulletin Board Subunit's Control Model, there are two kinds of controllers: reading devices (or simply Readers) and posting devices. The Reader is the unit that reads information from a board in an appropriate device to get some information about the device. The posting device is the unit that originated the information, and writes the information on an appropriate board in an appropriate device. The content of information is defined in each of the Board Type Specifications.

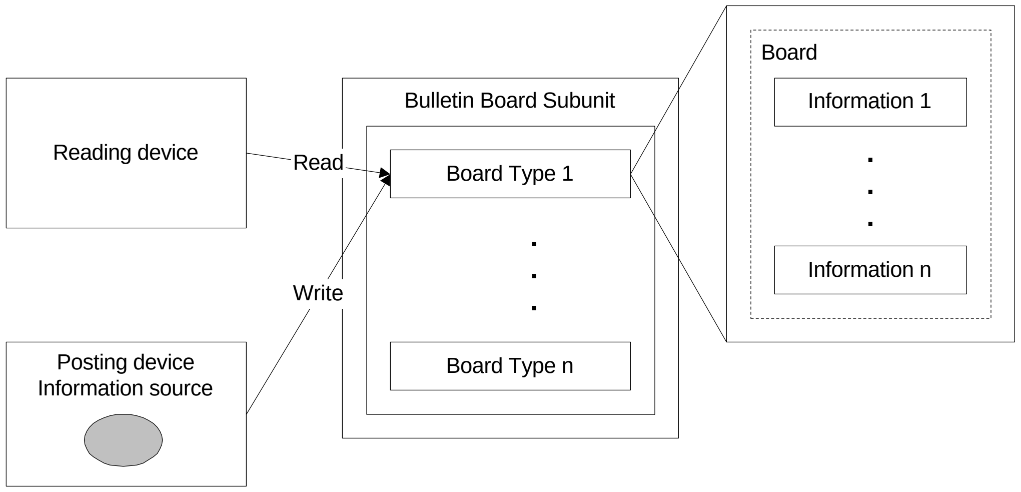

Figure 5.5 – Control model

Figure 5.5 shows three independent units. However, all or any combination of posting device, reading device, and Bulletin Board Subunit could be found in a single unit.

5.4.2 Creating, reading, writing, and deleting information

The operations that create information are executed using descriptor commands as defined in the Enhancements to the AV/C General Specification 3.0 version 1.0 [R2]. The operations that read, write and delete information are executed using descriptor commands as defined in the AV/C Digital Interface Command Set General Specification version 3.0 [R1]. The clauses that follow illustrate examples of how to perform these functions. They do not list all possible cases, and all commands may not be required.

5.4.2.1 Creating information

If a board has a Write Enabled list, the posting device can post (create) new entries in the list. When the posting device posts new information, it sets the Object ID with its own GUID and a unique record ID within the posting device itself. Creating entries follows the process below:

1) The posting device checks the open status of the list descriptor using the OPEN DESCRIPTOR status command with the appropriate *list_ID*. A successful response (status = 00_{16}, 01_{16}) ensures that the descriptor can be opened for write.

   NOTE — Even if the response from the status command shows that the descriptor can be opened, another controller could interrupt the ensuing OPEN DESCRIPTOR Control command by issuing its own OPEN DESCRIOPTOR Control command, subfunction = WRITE OPEN. Thus, the previous command does not always ensure that the descriptor can be opened.

2) The posting device opens the list descriptor for Write access using the OPEN DESCRIPTOR control command. Once the list descriptor is opened, all entry descriptors within the list descriptor are accessible.

3) At this time, the posting device may read shared information in the list to check for any board specific information it may need prior to posing new information.

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

4) The posting device uses the CREATE DESCRIPTOR control command on the opened list descriptor to create an entry descriptor. The posting device specifies the list descriptor *list_ID* and position in the "where" operand, and the *entry_type* in the "what" operand. Upon successful completion of this command, the Bulletin Board will have created an entry descriptor at the specified position with: *Object_ID* = 0 assigned by the Bulletin Board, and no *entry_specific_information* (*entry_specific_information length* = 0).
5) The posting device writes the *entry_specific_information* data with the WRITE DESCRIPTOR control command, with the *partial_replace* subfunction.
6) The posting device writes the entry descriptor's *object_ID* in one transaction using the WRITE DESCRIPTOR control command with the *partial_replace* subfunction.
7) To write multiple objects, the controller repeats the above three steps.
8) The posting device closes the opened list descriptor with the OPEN DESCRIPTOR control command, with the *close* subfunction.
9) When the Bulletin Board Subunit closes the list in response to the OPEN DESCRIPTOR command above, it checks for any entries with *Object_ID* = 0. If any exist, the Bulletin Board Subunit deletes them.

5.4.2.2 Reading Information

Any controller can read information posted by any device. The following process is followed when a posting device reads an information entry descriptor.

1) The posting device checks the open status of the list descriptor using the OPEN DESCRIPTOR status command with the appropriate *list_ID*. A successful response (status = 00_{16}, 01_{16}) ensures that the descriptor can be opened for read.
2) The posting device opens the list descriptor for Read access using the OPEN DESCRIPTOR control command. Once the list descriptor is opened, all entry descriptors within the list descriptor are accessible.
3) The posting device reads the entry descriptor by *object_ID* or by position in the list using the READ DESCRIPTOR control command.
4) The posting device closes the opened list descriptor with the OPEN DESCRIPTOR control command, with the *close* subfunction.

5.4.2.3 Writing Information

If a board has a Write Enabled list, the posting device can write or modify its own information. The following process is followed when a posting device writes to an existing information entry descriptor.

NOTE — Only the *object_ID* and *entry_specific_information* fields are written. Since the *entry_specific_information* size may change, the information entry's *size_of_entry_specific_information* field, and *descriptor_length* field, and the information list's *descriptor_length* field shall be updated by the Bulletin Board Subunit.

1) The posting device checks the open status of the list descriptor using the OPEN DESCRIPTOR status command with the appropriate *list_ID*. A successful response (status = 00_{16}, 01_{16}) ensures that the descriptor can be opened for write.
2) The posting device opens the list descriptor for Write access using the OPEN DESCRIPTOR control command. Once the list descriptor is opened, all entry descriptors within the list descriptor are accessible.

3) The posting device writes the *entry_specific_information* data using the WRITE DESCRIPTOR control command, the *partial_replace* subfunction.

4) The posting device closes the opened list descriptor with the OPEN DESCRIPTOR control command, with the *close* subfunction.

5.4.2.4 Deleting Information

The posting device can delete its own information. Normally, the posting device should delete its own information entry if it is no longer used. Any controller, however, can delete any object entry whose *Object_ID* = 0. Additional conditions for deletion may be defined in each Board Type Specification.

The following process is followed when a controller deletes an existing information entry descriptor.

1) The posting device checks the open status of the list descriptor using the OPEN DESCRIPTOR status command with the appropriate *list_ID*. A successful response (status = 00_{16}, 01_{16}) ensures that the descriptor can be opened for write.

2) The posting device opens the list descriptor for write access using the OPEN DESCRIPTOR control command. Once the list descriptor is opened, all entry descriptors within the list descriptor are considered opened.

3) The posting device deletes the entry descriptor using the WRITE DESCRIPTOR control command, with the *delete* subfunction.

4) The posting device closes the opened list descriptor with OPEN DESCRIPTOR control command, with the *close* subfunction.

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

6. Subunit Identifier Descriptor

The Bulletin Board Subunit's Subunit Identifier Descriptor describes the characteristics of the supported board types and information about the Bulletin Board Subunit itself. The Subunit Identifier Descriptor shall be managed by the Bulletin Board Subunit.

Figure 6.1 is the General Subunit Identifier Descriptor. The descriptor is fully explained in the AV/C Digital Interface Command Set General Specification version 3.0 [R1]. Fields with bold styles have some Bulletin Board Subunit specific comments in the clauses that follow.

Address	Contents
00 00_{16}	descriptor_length
00 01_{16}	
00 02_{16}	generation_ID
00 03_{16}	**size_of_list_ID**
00 04_{16}	**size_of_object_ID**
00 05_{16}	**size_of_object_position**
00 06_{16}	number_of_root_object_lists (n)
00 07_{16}	
00 08_{16}	**root_object_list_id_0**
:	
:	**:**
:	**root_object_list_id_n-1**
:	
XX XX_{16}	subunit_dependent_information_length
XX XX_{16} + 1	
:	**subunit_dependent_info (see clause 6.1.1)**
:	
:	
YY YY_{16}	manufacturer_dependent_length
YY YY_{16} + 1	
:	manufacturer_dependent_information
:	
:	

Figure 6.1 – The general Subunit Identifier Descriptor

6.1 Bulletin Board Subunit Identifier Descriptor field values

The Bulletin Board Subunit specifies the following values:

size_of_list_ID: The value of *size_of_list_ID* shall be 02_{16}. All lists in this subunit shall use 2 bytes for their list ID values.

size_of_object_ID: The value of *size_of_object_ID* shall be $0C_{16}$. All objects shall use 12 bytes for their Object ID values.

size_of_object_position: The *size_of_object_position* shall be 02_{16}. All such reference used with the subunit shall use 2 bytes for the position reference.

root_object_list_ID: The *root_object_list_id_x* fields indicate the ID values for the associated object lists (Board Lists or Information Lists). The following table illustrates the *list_ID* value with the board specific assignments. The lsb corresponds to the board type as listed in Table 5.1.

Table 6.1 – root_object_list_ID Value Assignment

Value	List definition
1001_{16}	Resource Schedule List
$1002\text{-}10FF_{16}$	reserved

6.1.1 subunit_dependent_information fields

The Bulletin Board Subunit shall have the following *subunit_dependent_information* fields.

Address offset	Contents
00_{16}	non_info_block_fields_length
01_{16}	
02_{16}	bulletin_board_subunit_version
03_{16}	number_of_supported_board_types (n)
04_{16}	supported_board_type_specific_information_length[0]
05_{16}	
06_{16}	supported_board_type_specific_information[0] (see clause 6.1.2)
:	
:	
:	:
:	supported_board_type_specific_information_length[n-1]
:	
:	supported_board_type_specific_information[n-1] (see clause 6.1.2)
:	
:	
:	optional info blocks for future expansion
:	
:	

Figure 6.2 – subunit_dependent_information fields

6.1.1.1 subunit_dependent_information field values

non_info_block_fields_length: The *non_info_block_fields_length* specifies the size, in bytes, of the non-info block fields through the *supported_board_type_specific_information[n-1]*. The value of this field does not include the length filed itself.

Controllers should be prepared to find any number of information blocks following this field, in case the *subunit_dependent_information_length* field needs to be expanded in the future. Controllers can easily determine if any info blocks exist here by comparing the *subunit_dependent_information_length* and *non_info_block_fields_length* fields.

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

If the following formula is true:

subunit_dependent_information_length > (*non_info_block_fields_length* + 2)

then info blocks exist in this structure.

bulletin_board_subunit_version: The b*ulletin_board_subunit_version* field indicates the version number of the Bulletin Board Subunit command specification that this Bulletin Board Subunit conforms to.

Table 6.2 – Bulletin Board Subunit version value assignment

Value	Meaning
00_{16} – $0F_{16}$	Reserved
10_{16}	Version 1.0 of the Bulletin Board Subunit specification
11_{16} – FF_{16}	Reserved for future specification

number_of_support_board_types: The *number_of_supported_board_types* field contains the number of different types of Boards supported by this subunit.

supported_board_type_specific_information_length:
The *supported_board_type_specific_information_length* field specifies the size, in bytes, of the *supported_board_type_specific_information*. The value of this field does not include the length field itself.

6.1.2 supported_board_type_specific_information fields

The *supported_board_type_specific_information* fields describe the characteristics of the supported board.

Address Offset	Contents
00_{16}	supported_board_type
01_{16}	supported_board_type_version
02_{16}	implementation_profile_ID
03_{16}	supported_board_type_dependent_information_length
04_{16}	
05_{16}	supported_board_type_dependent_information
:	
:	

Figure 6.3 – supported_board_type_specific_information fields

6.1.2.1 supported_board_type_specific_information field values

supported_board_type: The *supported_board_type* field identifies the type of board. Refer to Table 5.1 for existing values for this field.

supported_board_type_version: The *supported_board_type_version* field indicates the version number of each Bulletin Board Type Specification. Refer to the board type specification for more information about the value of this field.

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

implementation_profile_ID: The *implementation_profile_ID* field indicates the profile ID version for this board type. Please refer to the board type specification for more information about the value of this field.

supported_board_type_dependent_information_length:
The *supported_board_type_dependent_information_length* field contains the number of bytes used by the *supported_board_type_dependent_information* fields.

supported_board_type_dependent_information: The *supported_board_type_dependent_information* fields contain information that is specific to each board type specification. For details, please refer to the appropriate board type specification.

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

7. Board List Descriptor

This section defines the Board List Descriptor for the Bulletin Board Subunit.

The Board List Descriptor and its associated entries shall be managed by the Bulletin Board Subunit.

Figure 7.1 is the General Object List Descriptor that is used for the Board List Descriptor. The descriptor is fully explained in the AV/C Digital Interface Command Set General Specification version 3.0 [R1].

The Bold style's fields have some Bulletin Board Subunit specific comments in the clauses that follow.

Address_offset	Contents
$00\ 00_{16}$	descriptor_length
$00\ 01_{16}$	
$00\ 02_{16}$	**list_type**
$00\ 03_{16}$	**attributes**
$00\ 04_{16}$	size_of_list_specific_information
$00\ 05_{16}$	
$00\ 06_{16}$	**list_specific_information (see clause 7.1.1)**
:	
:	
:	number_of_entries(n)
:	
:	**object_entry[0]**
:	
:	
:	**:**
:	**object_entry[n-1]**
:	
:	

Figure 7.1 – The General Object List descriptor

7.1 Board List Descriptor field values

list_type: The Board List Descriptor *list_type* values are shown in Table 5.2.

attributes: Setting the *attributes* field is described in the AV/C Digital Interface Command Set General Specification version 3.0 [R1]. For a Board List Descriptor, the *has_Object_ID* attribute shall be set to 0.

7.1.1 list_specific_information

The *list_specific_information* fields depend on the value of *list_type*.

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

7.1.1.1 Read Only board list_specific_information fields

The following figure shows the *list_specific_information* fields for a Read Only Board List (*list_type* = 80_{16}).

Address_offset	Contents
00_{16}	non_info_block_fields_length
01_{16}	
02_{16}	board_type
03_{16}	board_type_dependent_information_length
04_{16}	
05_{16}	
:	board_type_dependent_information
:	
:	
:	optional info blocks for future expansion
:	

Figure 7.2 – Read only list_specific_information fields

7.1.1.1.1 Read Only board list_specific_information fields values

non_info_block_fields_length: The *non_info_block_fields_length* field specifies the number of bytes for the following non-info block fields, which extends through the *board_type_dependent_information* fields area.

board_type: The *board_type* is the type of board, and corresponds to one of the supported board types as given in the Subunit Identifier Descriptor. See Table 5.1 – Existing board types for more information on supported board types.

board_type_dependent_information_length: The *board_type_dependent_information_length* field specifies the size, in bytes, of the *board_type_dependent_information*. The value of this field does not include the length field itself.

board_type_dependent_information: The *board_type_dependent_information* fields contain board type specific information. For details, please refer to the appropriate board type specification document.

Optional info blocks for future expansion: The optional information blocks are to allow for future expansion and presently contain no information. These fields may be defined in the future.

7.1.1.2 Write Enabled board list_specific_information fields

The following figure is the *list_specific_information* fields for a Write Enabled Board List (*list_type* = 81_{16}).

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

Address_offset	Contents
00_{16}	non_info_block_fields_length
01_{16}	
02_{16}	board_type
03_{16}	object_list_maximum_size
04_{16}	
05_{16}	object_entries_maximum_number
06_{16}	
07_{16}	object_entry_maximum_size
08_{16}	
09_{16}	board_type_dependent_information_length
$0A_{16}$	
$0B_{16}$	board_type_dependent_information
:	
:	
:	optional info blocks for future expansion
:	
:	

Figure 7.3 – Write enabled list_specific_information fields for Board List Descriptor

7.1.1.2.1 Write Enabled board list_specific_information field values

non_info_block_fields_length: The *non_info_block_fields_length* field specifies the number of bytes for the following non info block fields, which extends through the *board_type_dependent_information* fields area. The value of this field does not include the length field itself.

board_type: The *board_type* is the type of board, and corresponds to one of the supported board types as given in the Subunit Identifier Descriptor. See Table 5.1 – Existing board types for more information on supported board types.

object_list_maximum_size: The *object_list_maximum_size* field indicates the maximum size of the object list. If the subunit's object list length is limited, the subunit sets this field to the limitation value. If the target doesn't specify a limitation, it sets this field to the value 0000_{16}.

object_entries_maximum_number: The *object_entries_maximum_number* field indicates the maximum number of object entries in the list. If the number of object entries is limited, the subunit sets this field to the limitation value. If the target doesn't specify a limitation, it sets this field to the value 0000_{16}.

object_entry_maximum_size: The *object_entry_maximum_size* field indicates the size of the object entry. If the subunit limits the length of the object entry, the subunit sets this field to the limitation value. If the target doesn't specify a limitation, it sets this field to the value 0000_{16}.

These three fields are useful for the controller to know the capacity of the object list or object entry. If each value for these three fields are all not 0, the Target device shall set these fields without contradiction using the following formula,

object_list_maximum_size >= *object_entries_maximum_number* X *object_entry_maximum_size* + overhead of the list

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

Where,

The overhead of the list includes all the fields in the Board List Descriptor that appear prior to the object entries. (The fields are *descriptor_ length* field, *list_type* field, *attributes* field, *size_of_list_specific_information* field, *list_specific_information* field, *number_of_entries* field)

board_type_dependent_information_length: The *board_type_dependent_information_length* field specifies the size, in bytes, of the *board_type_dependent_information*. The value of this field does not include the length field itself.

board_type_dependent_information: The *board_type_dependent_information* fields contain board type specific information. For details, please refer to the appropriate board type specification document.

Optional info blocks for future expansion: The optional information blocks are to allow for future expansion and presently contain no information. These fields may be defined in the future.

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

8. Board Entry Descriptor

The general Object Entry Descriptor is used for Board Entry Descriptors. The Object Entry Descriptor is fully explained in the AV/C Digital Interface Command Set General Specification version 3.0 [R1]. This section defines the Bulletin Board Subunit specific data fields for the Board Entry Descriptor.

The Bulletin Board Subunit shall manage all fields of the Board Entry Descriptor except the *entry_specific_information* fields.

Figure 8.1 shows the Board Entry Descriptor. The fields in bold type have some Bulletin Board Subunit specific comments in the clauses that follow.

Address_offset	Contents
00 00_{16}	descriptor_length
00 01_{16}	
00 02_{16}	**entry_type**
00 03_{16}	**attributes**
00 04_{16}	**child_list_ID**
00 05_{16}	
00 06_{16}	size_of_entry_specific_information
00 07_{16}	
00 08_{16}	**entry_specific_information**
:	
:	

Figure 8.1 – The Board Entry Descriptor

NOTE —The Board Entry Descriptor does not contain an *object_ID*.

8.1 Board Entry Descriptor field values

entry_type: The *entry_type* for the Board Entry Descriptor shall be 80_{16} = “Board”.

attributes: The attributes field shall have *has_child_ID* set to 1. Other attributes are defined in the AV/C Digital Interface Command Set General Specification version 3.0 [R1].

child_list_ID: The *child_list_ID* fields for the Board Entry Descriptor shall be managed by the Bulletin Board subunit. Refer to Table 5.4 for the range of values that the *child_list_ID* can have.

entry_specific_information: The *entry_specific_information* fields are defined in each board type specification. If the list is write enabled, a controller may write to these fields according to the specific board type specification.

9. Information List Descriptor

This section defines the Information List Descriptor for the Bulletin Board Subunit.

The Information List Descriptor shall be managed mostly by the Bulletin Board Subunit. However, the controller may write to the *board_type_dependent* and *optional info block* fields if the list is write enabled.

Figure 7.1 is the General Object List Descriptor that is used for the Information List Descriptors. The descriptor is fully explained in the AV/C Digital Interface Command Set General Specification version 3.0 [R1].

The fields in bold type have some Bulletin Board Subunit specific comments in the clauses that follow.

Address_offset	Contents
00 00_{16}	descriptor_length
00 01_{16}	
00 02_{16}	**list_type**
00 03_{16}	**attributes**
00 04_{16}	size_of_list_specific_information
00 05_{16}	
00 06_{16}	**list_specific_information (see clause 9.1.1)**
:	
:	
XX XX_{16}	number_of_entries(n)
XX XX_{16} + 1	
:	**object_entry[0]**
:	
:	
:	**:**
:	**object_entry[n-1]**
:	
:	

Figure 9.1 – The General Object List descriptor

9.1 Information List Descriptor field values

list_type: The Information List Descriptor *list_type* values are shown in Table 5.2.

attributes: Setting the *attributes* field is described in the AV/C Digital Interface Command Set General Specification version 3.0 [R1]. For Information List Descriptors, the "*has_object_ID*" attribute shall be set to 1. All objects in the Bulletin Board Subunit have an Object ID that specifies the posting device with the posting device GUID as the Object ID.

9.1.1 list_specific_information

The *list_specific_information* fields depend on the value of *list_type*.

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

9.1.1.1 Read Only list_specific_information fields

The following figure shows the *list_specific_information* fields for a Read Only Information List (*list_type* = 82_{16}).

Address_offset	Contents
00_{16}	non_info_block_fields_length
01_{16}	
02_{16}	board_type
03_{16}	board_type_dependent_information_length
04_{16}	
05_{16}	board_type_dependent_information
:	
:	
:	optional info blocks for future expansion
:	
:	

Figure 9.2 – Read only list_specific_information fields

9.1.1.1.1 Read Only list_specific_information field values

non_info_block_fields_length: The *non_info_block_fields_length* field specifies the number of bytes for the following non-info block fields, which extends through the *board_type_dependent_information* fields area. The value of this field does not include the length field itself.

board_type: The *board_type* is the type of board, and corresponds to one of the supported board types as given in the Subunit Identifier Descriptor. See Table 5.1 – Existing board types for more information on supported board types.

board_type_dependent_information_length: The *board_type_dependent_information_length* field specifies the size, in bytes, of the *board_type_dependent_information*. The value of this field does not include the length field itself.

board_type_dependent_information: The *board_type_dependent_information* fields contain board type specific information. For details, please refer to the appropriate board type specification document.

Optional info blocks for future expansion: The optional information blocks are to allow for future expansion and presently contain no information. This field may be defined in the future.

9.1.1.2 Write Enabled list_specific_information fields

The following figure is the *list_specific_information* fields for a Write Enabled Information List (*list_type* = 83_{16}). A controller can update the Fields in gray according to the specific board type specification.

Address_offset	Contents
00_{16}	non_info_block_fields_length
01_{16}	
02_{16}	board_type
03_{16}	object_list_maximum_size
04_{16}	
05_{16}	object_entries_maximum_number
06_{16}	
07_{16}	object_entry_maximum_size
08_{16}	
09_{16}	board_type_dependent_information_length
$0A_{16}$	
$0B_{16}$	board_type_dependent_information
:	
:	
:	optional info blocks for future expansion
:	
:	

Figure 9.3 – Write enabled list_specific_information fields for the Information List Descriptor

9.1.1.2.1 Write Enabled list_specific_information field values

non_info_block_fields_length: The *non_info_block_fields_length* field specifies the number of bytes for the following non info block fields, which extends through the *board_type_dependent_information* fields area. The value of this field does not include the length field itself.

object_list_maximum_size: The *object_list_maximum_size* field indicates the maximum size of the object list. If the subunit's object list length is limited, the subunit sets this field to the limitation value. If the target doesn't specify a limitation, it sets this field to the value 0000_{16}.

object_entries_maximum_number: The *object_entries_maximum_number* field indicates the maximum number of object entries in the list. If the number of object entries is limited, the subunit sets this field to the limitation value. If the target doesn't specify a limitation, it sets this field to the value 0000_{16}.

object_entry_maximum_size: The *object_entry_maximum_size* field indicates the size of the object entry. If the subunit limits the length of the object entry, the subunit sets this field to the limitation value. If the target doesn't specify a limitation, it sets this field to the value 0000_{16}.

The above three fields are useful for the controller to know the capacity of the object list or object entry. If each value for these three fields are all not 0, the Target device shall set these fields without contradiction using the following formula,

> *object_list_maximum_size* >= *object_entries_maximum_number* X *object_entry_maximum_size* + overhead of the list
>
> Where,

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

The overhead of the list includes all the fields in the Information List Descriptor that appear prior to the object entries.

board_type: The *board_type* is the type of board, and corresponds to one of the supported board types as given in the Subunit Identifier Descriptor. See Table 5.1 – Existing board types for more information on supported board types.

board_type_dependent_information_length: The *board_type_dependent_information_length* field specifies the size, in bytes, of the *board_type_dependent_information*. The value of this field does not include the length field itself.

board_type_dependent_information: The *board_type_dependent_information* fields contain board type specific information. For details, please refer to the appropriate board type specification document.

Optional info blocks for future expansion: The optional information blocks are to allow for future expansion and presently contain no information. This field may be defined in the future.

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

10. Information Entry Descriptor

The general Object Entry Descriptor is used for Information Entry Descriptors. The Object Entry Descriptor is fully explained in the AV/C Digital Interface Command Set General Specification version 3.0 [R1]. This section defines the Bulletin Board Subunit specific data fields for the Information Entry Descriptor.

If the list type is Write Enabled, the controller writes the *object_ID* fields and *entry_specific_information* fields. The other fields shall be managed by the Bulletin Board Subunit. If a controller tries to write to any fields other than the *object_ID* and *entry_specific_information* fields, the operation shall be rejected.

Figure 10.1 shows the Information Entry Descriptor.

Address_offset	Contents
00 00_{16}	descriptor_length
00 01_{16}	
00 02_{16}	**entry_type**
00 03_{16}	attributes
00 04_{16}	**object_ID** (12 bytes, see clause 10.1.1)
:	
00 $0F_{16}$	
00 10_{16}	size_of_entry_specific_information
00 11_{16}	
00 12_{16}	**entry_specific_information**
:	
:	

Figure 10.1 – The Information Entry Descriptor

Note that Information Entry Descriptors may contain *child_list_IDs*.

10.1 Information Entry Descriptor field values

entry_type: The *entry_type* for the Board Entry Descriptor shall be 81_{16} = “Information”.

entry_specific_information: The *entry_specific_information* fields are defined in each board type specification.

10.1.1 Object_ID fields

The *object_ID* fields in the Information Entry Descriptor are composed of the *posting_device_GUID* and the *record_ID* and are specified as follows:

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

Address_offset	Contents
00_{16}	posting_device_GUID
01_{16}	
02_{16}	
03_{16}	
04_{16}	
05_{16}	
06_{16}	
07_{16}	
08_{16}	record_ID
09_{16}	
$0A_{16}$	
$0B_{16}$	

Figure 10.2 – object_ID fields

10.1.1.1 object_ID field values

posting_device_GUID: The *posting_device_GUID* is the Global Unique ID of a device that has the information source. When the entry is created using the CREATE DESCRIPTOR command, the Bulletin Board Subunit shall set the *posting_device_GUID* to all 0s.

record_ID: The *record_ID* is an ID assigned by the posting device. A posting device shall write a unique *record_ID* within each board type.

The entry's object_ID (which consists of the posting_device_GUID and record_ID) shall be unique for each entry within a board type. For example, even though posting devices A and B may write the same *record_ID*, their *posting_device_GUID* is different, ensuring that the *Object_ID* is unique.

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

11. Bulletin Board Subunit commands

11.1 Bulletin Board Subunit type

Bulletin Board Subunit shall be identified by a *subunit_type* of $0A_{16}$.

11.2 Commands support level

There are no Bulletin Board Subunit-specific commands defined.

This section defines the required commands for the Bulletin Board Subunit General specification.

The marks mean as follows;

— "M" means that the command is mandatory and is required.

— "M*" means that the command is mandatory and is required if the list type is write enabled.

— "O" means that the command is optional.

— "NS" means that the command is not supported.

— "B" means that the command may be defined as mandatory in a board type specification.

The device shall implement all commands which are specified as mandatory in Bulletin Board Subunit General Specification and in supported Board Type Specifications.

Table 11.1 – Bulletin Board Subunit commands support level

opcode/operand		Support level
OPEN DESCRIPTOR (control)		
subfunction	CLOSE	M
	READ OPEN	M
	WRITE OPEN	M*
descriptor_identifier	subunit Identifier	M
	an object list specified by ID	M
	an object list specified by list_type	NS
	an object entry position reference	O
	an object ID reference	O

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

opcode/operand			Support level
OPEN DESCRIPTOR (status)			
descriptor_identifier	subunit Identifier		M
	an object list specified by ID		M
	an object list specified by list_type		NS
	an object entry position reference		O
	an object ID reference		O
OPEN DESCRIPTOR (notify)			
descriptor_identifier	subunit Identifier		O
	an object list specified by ID		O
	an object list specified by list_type		NS
	an object entry position reference		O
	an object ID reference		O
READ DESCRIPTOR (control)			
descriptor_identifier	subunit Identifier		M
	an object list specified by ID		M
	an object list specified by list_type		NS
	an object entry position reference		M
	an object ID reference		M
WRITE DESCRIPTOR (control)			
subfunction	change		M*
	replace		NS
	insert		NS
	delete		M*
	partial replace		M*
descriptor_identifier	subunit Identifier		NS
	an object list specified by ID		B
	an object list specified by list_type		NS
	an object entry position reference		M*
	an object ID reference		M*
group_tag	first, continue, last		O
WRITE DESCRIPTOR (status)			
descriptor_identifier	subunit Identifier		NS
	an object list specified by ID		B
	an object list specified by list_type		NS
	an object entry position reference		M*
	an object ID reference		M*
CREATE DESCRIPTOR (control)			
subfunction_1	create a new object and child list		B
subfunction_1	create a new descriptor		
	descriptor_identifier	create a root list	NS
		create a child list	B
		create an object	B

opcode/operand		Support level
OPEN INFO BLOCK (control)		
subfunction	CLOSE	O
	READ OPEN	O
	WRITE OPEN	O
descriptor_identifier	subunit Identifier	O
	an object list specified by ID	O
	an object list specified by list_type	NS
	an object entry position reference	O
	an object ID reference	O
info_block_identifier	by its type and instance count	O
	by position in the container structure	O
OPEN INFO BLOCK (status)		
descriptor_identifier	subunit Identifier	O
	an object list specified by ID	O
	an object list specified by list_type	NS
	an object entry position reference	O
	an object ID reference	O
info_block_identifier	by its type and instance count	O
	by position in the container structure	O
READ INFO BLOCK (control)		
descriptor_identifier	subunit Identifier	B
	an object list specified by ID	B
	an object list specified by list_type	NS
	an object entry position reference	B
	an object ID reference	B
info_block_identifier	by its type and instance count	B
	by position in the container structure	B
WRITE INFO BLOCK (control)		
subfunction	partial_replace	B
descriptor_identifier	subunit Identifier	NS
	an object list specified by ID	B
	an object list specified by list_type	NS
	an object entry position reference	B
	an object ID reference	B
info_block_identifier	by its type and instance count	B
	by position in the container structure	B
group_tag	first, continue, last	O
RESERVE (control/status/notify)		
		NS
PLUG INFO (status)		
		NS

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.
1394 TRADE ASSOCIATION THE MULTIMEDIA CONNECTION

11.3 The behavior in the RESERVED situation

The Bulletin Board Subunit does not support the RESERVE command.

When the RESERVE command is addressed to the unit, the reservation has no effect on its Bulletin Board Subunit.

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

Annex A: Bulletin Board Descriptors High Level View (informative)

A.1 Subunit Identifier Descriptor high-level view

The following figure is the Bulletin Board Subunit's Subunit Identifier Descriptor in a high-level view. This view includes the *Subunit independent information* fields and *Subunit board type specific information* fields. Note that the field sizes are not shown.

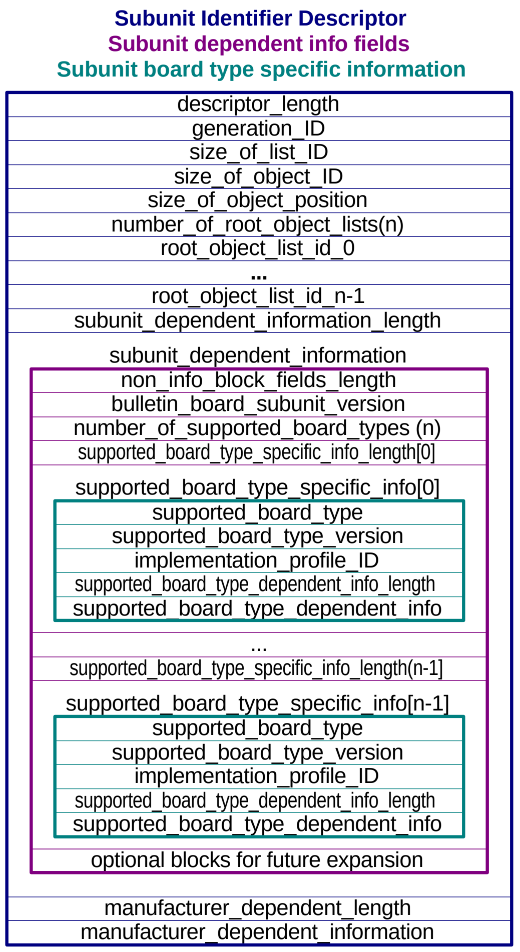
Subunit Identifier Descriptor
Subunit dependent info fields
Subunit board type specific information
descriptor_length
generation_ID
size_of_list_ID
size_of_object_ID
size_of_object_position
number_of_root_object_lists(n)
root_object_list_id_0
...
root_object_list_id_n-1
subunit_dependent_information_length
subunit_dependent_information
non_info_block_fields_length
bulletin_board_subunit_version
number_of_supported_board_types (n)
supported_board_type_specific_info_length[0]
supported_board_type_specific_info[0]
supported_board_type
supported_board_type_version
implementation_profile_ID
supported_board_type_dependent_info_length
supported_board_type_dependent_info
...
supported_board_type_specific_info_length(n-1]
supported_board_type_specific_info[n-1]
supported_board_type
supported_board_type_version
implementation_profile_ID
supported_board_type_dependent_info_length
supported_board_type_dependent_info
optional blocks for future expansion
manufacturer_dependent_length
manufacturer_dependent_information

Figure B.1 – Subunit Identifier Descriptor high level view

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.

A.2 Board List Descriptor high-level view

The following figure is the Bulletin Board Subunit's Board List Descriptor in a high-level view. This view includes the *List specific information* fields and *Board Entry Descriptor* fields. Note that the field sizes are not shown.

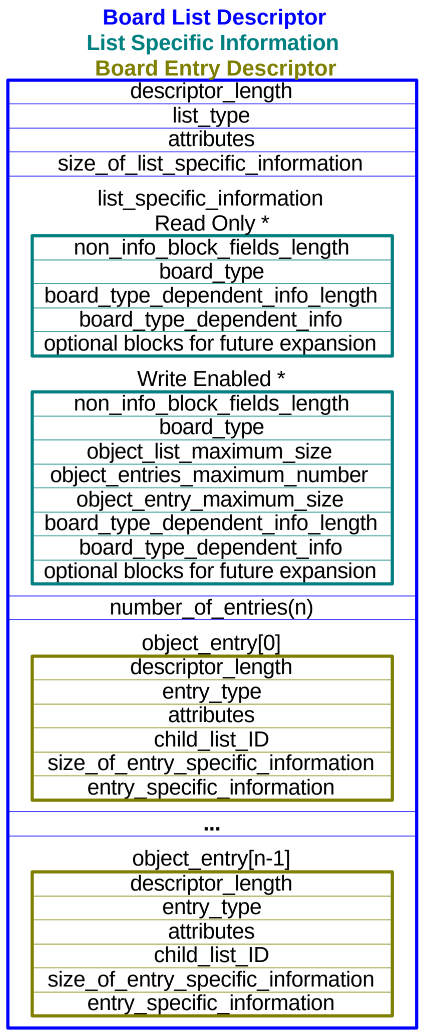
Board List Descriptor
List Specific Information
Board Entry Descriptor
descriptor_length
list_type
attributes
size_of_list_specific_information
list_specific_information
Read Only *
non_info_block_fields_length
board_type
board_type_dependent_info_length
board_type_dependent_info
optional blocks for future expansion
Write Enabled *
non_info_block_fields_length
board_type
object_list_maximum_size
object_entries_maximum_number
object_entry_maximum_size
board_type_dependent_info_length
board_type_dependent_info
optional blocks for future expansion
number_of_entries(n)
object_entry[0]
descriptor_length
entry_type
attributes
child_list_ID
size_of_entry_specific_information
entry_specific_information
...
object_entry[n-1]
descriptor_length
entry_type
attributes
child_list_ID
size_of_entry_specific_information
entry_specific_information

Figure B.2 – Board List Descriptor high level view

*Either Read Only or Write Enabled blocks can exist in the list specific information

A.3 Information List Descriptor high-level view

The following figure is the Bulletin Board Subunit's Information List Descriptor in a high-level view. This view includes the *List specific information* fields, *Information Entry Descriptor* fields, and *Object_ID* fields. Note that the field sizes are not shown.

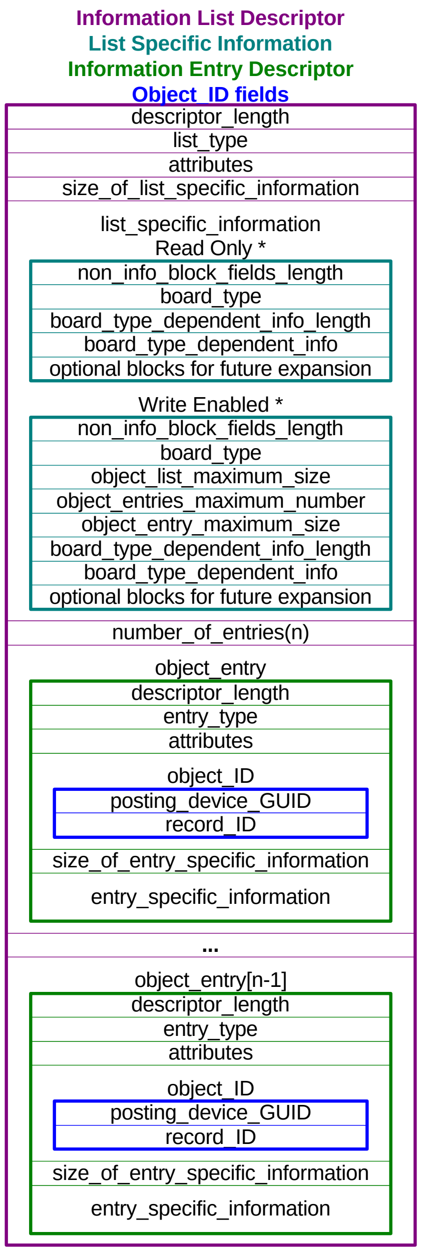
Information List Descriptor
List Specific Information
Information Entry Descriptor
Object_ID fields
descriptor_length
list_type
attributes
size_of_list_specific_information
list_specific_information
Read Only *
non_info_block_fields_length
board_type
board_type_dependent_info_length
board_type_dependent_info
optional blocks for future expansion
Write Enabled *
non_info_block_fields_length
board_type
object_list_maximum_size
object_entries_maximum_number
object_entry_maximum_size
board_type_dependent_info_length
board_type_dependent_info
optional blocks for future expansion
number_of_entries(n)
object_entry
descriptor_length
entry_type
attributes
object_ID
posting_device_GUID
record_ID
size_of_entry_specific_information
entry_specific_information
...
object_entry[n-1]
descriptor_length
entry_type
attributes
object_ID
posting_device_GUID
record_ID
size_of_entry_specific_information
entry_specific_information

Figure B.3 – Information List Descriptor high level view

*Either Read Only or Write Enabled blocks can exist in the list specific information

Copyright © 1999, 1394 Trade Association. All rights reserved.
This is an unapproved TA Specification Draft, subject to change.