Secretariat 2024/08/25 00:08 DRAFT REVISED AES70-21-xxxx

STANDARDS AND INFORMATION DOCUMENTS

Call for comment on DRAFT AES standard for Audio applications of networks - Open Control Architecture - Part 22: Using AES70 to manage Milan™ media transport

This document was developed by a writing group of the Audio Engineering Society Standards Committee (AESSC) and has been prepared for comment according to AES policies and procedures. It has been brought to the attention of International Electrotechnical Commission Technical Committee 100. Existing international standards relating to the subject of this document were used and referenced throughout its development.

Address comments by E-mail to standards@aes.org, or by mail to the AESSC Secretariat, Audio Engineering Society, 697 Third Ave., Suite 405, New York NY 10017. **Only comments so addressed will be considered**. E-mail is preferred. **Comments that suggest changes must include proposed wording.** Comments shall be restricted to this document only. Send comments to other documents separately. Recipients of this document are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

This document will be approved by the AES after any adverse comment received within **six weeks** of the publication of this call on http://www.aes.org/standards/comments/, **2024-08-25**, has been resolved. Any person receiving this call first through the *JAES* distribution may inform the Secretariat immediately of an intention to comment within a month of this distribution.

Because this document is a draft and is subject to change, no portion of it shall be quoted in any publication without the written permission of the AES, and all published references to it must include a prominent warning that the draft will be changed and must not be used as a standard.

AES standard for audio applications of networks - Open Control Architecture - Part 22: Using AES70 to manage Milan™ media transport

Published by
Audio Engineering Society, Inc.

Abstract

AES70 is a suite of standards for control and monitoring of devices in professional media networks. This standard, *AES standard for Audio applications of networks - Open control architecture - Part 22: Using AES70 to manage Milan™ media transport* defines an application of the Core AES70 specification for managing Milan™ media transport connections, and related synchronization and clocking mechanisms.. Other standards in the AES70 suite specify control and monitoring repertoire, control protocols, and media transport management applications.

AES70 does not specify a media transport scheme. Rather, it is designed to operate with media transport schemes such as the one specified by Milan™.

AES70's intended range of use spans networks of all sizes. This includes mission-critical applications, high-security applications, IP and non-IP networks, and local and wide-area applications. AES70 can control real or virtual devices located on premises or hosted by cloud services. AES70 consumes little computing power and uses network bandwidth lightly.

AES70 architecture is network-agnostic. Current AES70 standards define protocols for use over IP networks and simple byte-stream networks, but other network types may readily be accommodated.

AES70 is based on the Open Control Architecture (OCA), originally developed by the OCA Alliance.

Audio Engineering Society Inc., 697 Third Avenue, Suite 405, New York, NY 10017, US.
www.aes.org/standards standards@aes.org

Foreword

This foreword is not part of this document, *AES standard for Audio applications of networks - Open control architecture - Part 22: Using AES70 to manage Milan™ media transport*.

The role of AES standards. An AES standard implies a consensus of those directly and materially affected by its scope and provisions and is intended as a guide to aid the manufacturer, the consumer, and the general public. Prior to the publication of an AES standard, all parties, including the general public, are given opportunities to comment on or object to any provision. Nevertheless, the existence of an AES standard shall not preclude anyone, whether or not he or she has approved the document, from manufacturing, marketing, purchasing, or using products, processes, or procedures not in agreement with the standard.

Patent rights. Attention is drawn to the possibility that some of the elements of this AES standard or information document may be the subject of patent rights. AES shall not be held responsible for identifying any or all such rights. Approval by the AES does not assume any liability to any patent owner, nor does it assume any obligation whatever to parties adopting the document.

Recipients of this document are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

Review and revision. This document is subject to periodic review and possible revision. Users are cautioned to obtain the latest edition.

AES70 Structure

The AES70 standard is a suite of standards, classified into two divisions. The ***Core Standards*** division, contains standards essential to all implementations of AES70; the ***Adaptation Standards*** division contains application-specific standards. This standard, *AES standard for Audio applications of networks - Open control architecture - Part 22: Using AES70 to manage Milan media transport*, is an Adaptation Standard.

AES70-22 Version history

This version, AES70-22-2024 is the first version of the Standard.

The Milan group members who developed the original draft that is the basis of this document were C. Ahrens, M. Lave, and, B. Escalona Espinosa. The AES task group members who produced this document from that draft were J. Berryman (chair), B. Escalona Espinosa, E. Hoehn, S. Jones, M. Lave, G. Linis, A. Rosen, S. Scott, P. Stevens, and P. Treleaven.

Morten Lave
Chair, AES SC-02-12, *Working Group on Audio Applications of Networks*
2024-08-24

Note on normative language

In AES standards documents, sentences containing the word "shall" are requirements for compliance with the document. Sentences containing the verb "should" are strong suggestions (recommendations). Sentences giving permission use the verb "may". Sentences expressing a possibility use the verb "can".

Contents

Tables

Figures

AES standard for audio applications of networks - Open Control Architecture - Part 22: Using AES70 to manage Milan™ media transport

0. Introduction

AES70 is a standards suite for media system control and monitoring via networks.

The AES70 standards suite has a number of separate parts. This document should be read in conjunction with [AES70-1], the framework standard, [AES70-2], the class structure standard, and [AES70-3], the protocol standard.

This document is a part of the 2024 version of the AES70 suite.

NOTE: Milan™ is a registered trademark of the AVnu Alliance, 3855 SW 153rd Dr., Beaverton, Oregon 97003, USA. In what follows, the text "Milan™" refers to this trademark; i.e. "Milan" in this standard should be read as "Milan®".

1. Scope

This Standard specifies the use of AES70 for managing Milan™ media transport connections and related synchronization and clocking mechanisms. Milan is an AVB profile for real-time audio transport, and is created and maintained by the AVnu Alliance.

The following use cases shall be supported:

1 A Controller uses AES70 to directly control stream connections of all Devices.

In this case, a Controller sends AES70 commands to both Talkers and Listeners to manage their Milan Endpoints and stream connection(s).

2 A Controller uses AES70 to control stream connections of the Talker(s), and out-of-scope means are used to control stream connections of the Listener(s).

In this case, a controller sends AES70 commands to an AES70-controlled Talker to manage its Milan Output Endpoints. Milan Input Endpoints and stream connections of Listeners are managed by out-of-scope means.

3 A Controller uses AES70 to control stream connections of the Listener(s), and out-of-scope means are used to control stream connections of the Talker(s).

In this case, a Controller sends AES70 commands to the AES70-controlled Listeners to manage their Milan Input Endpoints and stream connections. Milan Output Endpoints of the Talkers are managed by out-of-scope means.

In the above, *out-of-scope means* can refer to the front panel of the Device, or a Milan compliant Layer 2 Controller, for example.

2. Normative references

The following referenced documents are indispensable for the application of this standard:

AES70-1. *"AES70-1-2022: AES standard for audio applications of networks - Open Control Architecture - Part 1: Framework"*, Audio Engineering Society.

AES70-2. *"AES70-2-2022: AES standard for audio applications of networks - Open Control Architecture - Part 2: Class structure"*, Audio Engineering Society.

AES70-2A. *"AES70-2-2022, Annex A: AES standard for audio applications of networks - Open Control Architecture - Part 2: Class structure; Annex A (normative) UML Class Structure Definition"*, Audio Engineering Society.

ATDECC. *IEEE 1722.1-2021, "IEEE Standard for Device Discovery, Connection Management, and Control Protocol for Time-Sensitive Networking System".*

AVnu-Milan. *"Milan Specification"*, AVnu Pro Audio Technical Workgroup.

AVTP. *IEEE 1722-2016, "IEEE Standard for Layer 2 Transport Protocol for Time-Sensitive Applications in Bridged Local Area Networks".*

IEEE-802.1AS. *IEEE Std. 802.1AS-2011 "IEEE Standard for Local and Metropolitan Area Networks – Timing and Synchronization for Time-Sensitive Applications in Bridged Local Area Networks".*

IEEE-802.1BA. *IEEE 802.1BA-2011, "Audio Video Bridging (AVB) Systems".*

IEEE-802.1Q. *IEEE 802.1Q-2014, "Media Access Control (MAC) Bridges and Virtual Bridge Local Area Networks".*

A document reference will be identified in the text by the document name, shown above in bold, in square brackets. A part of a document will be identified by a suffix of the form (<section heading>), attached to the document name. For example, [AES70-1] represents the AES70-1-2022 standard, and [AES70-1(Media transport control)] represents the Media transport control chapter of it.

3. Terms, definitions, and abbreviations

See [AES70-1] or [AVnu-Milan] for most terms. In addition, this standard defines or redefines the following terms:

1. AAF
AVTP Audio Format, the stream format AVTP specifies for transport of audio samples.

2. Adaptation
formal specification of an AES70 application for managing Milan media transport.

3. ACMP
ATDECC Connection Management Protocol used by ATDECC (see [ATDECC]).

4. ADP
ATDECC Discovery Protocol (Milan term, see ATDECC(6)]).

5. AVB
layer-1 networking standard for the movement of time-sensitive traffic through Ethernets (see [IEEE-802.1BA]).

6. AVTP
layer-2 protocol standard that uses AVB for the low-latency transport of synchronous data. Synonym of "IEEE-1722" (see [AVTP]).

7. ATDECC
streaming data connection management model for AVTP. Synonym of "IEEE 1722.1" (see [ATDECC]).

8. CM4
connection management feature set of AES70-2024.

9. Core AES70
members of the AES70 standards family that define the fundamentals of AES70: AES70-1, AES70-2, AES70-3, and possible future standards. In the AES70 suite, Core standards have document identifiers with numeric suffixes less than 20.

10. CRF
Clock Reference Format, the stream format AVTP specifies for transport of clock reference data.

11. Device
device that is compliant with this Adaptation.

12. Endpoint
instance of the OcaMediaStreamEndpoint datatype (see Clause 7.3.6).

13. Entity
logical object within a Device that implements a set of descriptors that conform to the Milan Entity Model.

14. Entity Model
set of descriptors that can be instantiated to form an Entity.

15. Listener
Device that is capable of receiving and consuming one or more Streams. (see [AVnu-Milan(Glossary)]).

16. Mapping, Channel Mapping
association between a channel of an Audio Cluster and a channel of a Stream.

17. NAC, Network Application Control
Core AES70 networking model (see [AES70-1(Networking model)]).

18. OCA Port
data element defined by the OcaPort class that describes one input or output signal channel of the processing function that a Worker or Network Application object represents. ***Input OCA Port*** means an OCA Port that represents signal flow into the object; ***Output Oca Port*** means an OCA Port that represents signal flow out of the object.

19. Stream Format String
8-byte descriptor that specifies the format of a Stream, the structure of which is defined in [AVTP].

20. Session
binding of a single local Input Endpoint to an AVTP Stream originated remotely.

21. Session State Machine
state machine responsible for establishing the connection path from the Talker to the Listener.

22. Stream
unidirectional flow of AVTP frames (see [AVnu-Milan(Glossary)]).

23. Stream Mode
payload properties of an individual media Stream connection including framing format, sample format, sampling rate, channel count, and packet time.

24. Stream Reservation Protocol (SRP)
protocol for allocating network bandwidth to Streams (see [IEEE-802.1Q]).

25. Talker
Device that is capable of producing and transmitting one or more Streams (see [AVnu-Milan(Glossary)]).

4. Document conventions

In this standard:

1. Full definitions of variables (class properties, method parameters, etc.) are given in the form:
2. All document conventions defined in [AES70-1(Document conventions)] apply.

   <*datatype*> <variablename>
   e.g.
   OcaString Label

3. Classes and datatypes defined in this standard, as well as referenced AES70 classes and datatypes shall be named according to the rules specified by [AES70-2(Custom subclass naming)].

   This standard's prefix shall be "Milan".
   The names of classes and datatypes used in this standard shall begin with this prefix.
   Example:
   Original class defined by Core AES70: OcaMediaTransportSessionAgent
   Subclass defined by this standard: MilanOcaMediaTransportSessionAgent

4. Referenced Milan descriptor names, field names and terms are given in the Arial font, and colored green, e.g.:

   <fieldname> field in <descriptorname>
   e.g.
   port_flags field in STREAM_PORT_INPUT

   The specific green value used in this document is RGB(83,192,53) = 0x538135.

   Capitalization rules are as follows (these are the same as the rules in [AVnu-Milan]):

- Names of Milan Device elements are in mixed case (e.g. Entity), whereas names of the Milan descriptors that apply to those elements are in upper case (e.g. ENTITY).
- Names of fields and properties of Milan descriptors are in lower case (e.g. port_flags)
- Names of flag bits and enum options are in upper case (e.g. REDUNDANCY).

5. The document illustrates operation sequences using Universal Modeling Language (UML) sequence diagrams.

5. Adaptation elements

This standard specifies the Adaptation elements listed in this clause. Equipment compliant with this Adaptation shall adhere to its specifications for these elements.

5.1. Adaptation-specific properties

In various places throughout the AES70 class structure, classes and datatypes have properties named AdaptationData of datatype OcaAdaptationData. This datatype is a synonym for OcaBlob.

The purpose of AdaptationData properties is to store Adaptation-specific information. For this Adaptation, the formats of such properties shall be specified by Milan-specific datatypes defined by this standard. Adaptation data values shall be marshaled into their respective OcaAdaptationData properties according to these datatypes and the marshaling rules in [AES70-3(Marshaling)].

5.2. Adaptation-specific Control Class

As specified by [AES70-1], an Adaptation-specific Control Class is defined by defining a new control class that is a subclass of a Core AES70 class. This standard defines a single Adaptation-specific subclass as follows:

MilanOcaMediaTransportSessionAgent, Class ID **1.2.20.A.2200**

Where **A** is the Authority ID that identifies the Authority responsible for defining the nonstandard classes used in this Adaptation. **A** shall be as follows:

0xFFFF00. 0x000B5E

This Authority ID contains the 24-bit Company ID (CID) or Organization Unique Identifier (OUI) issued by the IEEE Registration Authority for proprietary classes defined by the AES, and follows the format defined in [AES70-1(Authority ID format)].

5.3. Adaptation identifier

AES70 requires each Adaptation to have a unique identifier. The identifier of this Adaptation shall be "OcaMilan".

6. Adaptation concept (informative)

This standard defines a mapping from the Milan streaming media connection model to the AES70 streaming media connection model. The two models are shown in Figure 1, and are described as follows:

1. **Milan**

   - The Milan connection management model is a specialization of ATDECC.
   - ATDECC (see [ATDECC]), also known as IEEE 1722.1, is a streaming data connection management model for AVTP.
   - AVTP (see [AVTP]), also known as IEEE 1722, is a layer-2 protocol standard for the low-latency transport of synchronous data. AVTP uses AVB packet transport.
   - AVB (see [IEEE-802.1BA]) is a layer-1 networking standard for the movement of time-sensitive traffic through Ethernets.

2. **AES70-22 (this standard)**

   - This standard defines a specialization of AES70 Connection Management 4 ("CM4").
   - CM4 (see [AES70-1(Media Transport Application Model)]) is a streaming data connection management model for AES70 Devices. CM4 is specified in [AES70-1(Media transport application model)].
   - CM4 has two sub-models. From top to bottom, these are:
     - The CM4 Network Application model, which provides control and monitoring access to media transport connections.
     - The CM4 Network Interface model, which provides control and monitoring access to basic data network services.

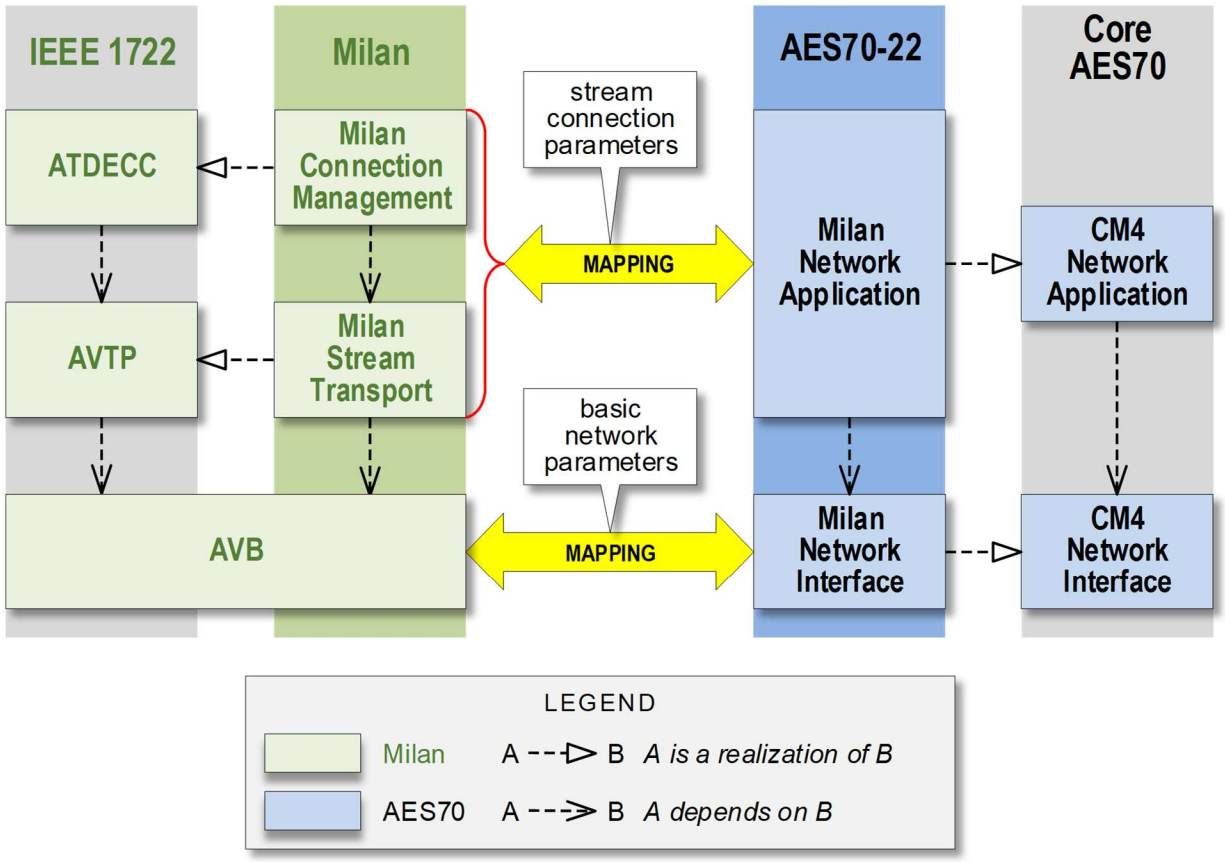

Figure 1. Adaptation concept

NOTE 1 Figure 1 uses UML arrow notation - see [AES70-1(Class diagram conventions)].

NOTE 2 CM4 is a specialization of a more abstract AES70 model named Network Application Control ("NAC"), but this detail is out of scope of AES70-22. Interested readers are referred to [AES70-1(Networking model)].

In what follows, Clause 7 normatively specifies the use of CM4 objects, properties, and methods for controlling and monitoring Milan connections. The text is organized into subclauses corresponding to the CM4 elements involved.

Annex A informatively presents an inverted view. The text is organized into subclauses corresponding to the Milan elements involved.

7. Adaptation details

The Adaptation specified by this document is a Milan specialization of CM4. Figure 2 illustrates the Adaptation's class subtree.

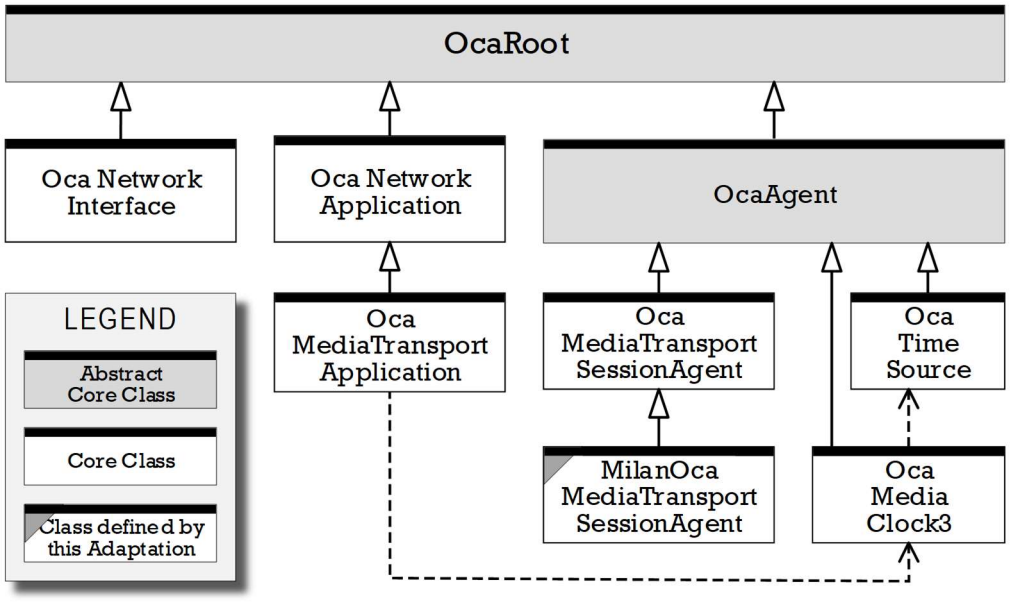

Figure 2. CM4 Milan Classes

In addition to classes, CM4 defines a number of significant datatypes. This Adaptation defines specialized versions of some of them. The complete set of classes and datatypes constitute the Milan NAC model, described in detail in the following clauses.

NOTE In ATDECC, descriptors typically have two names: an "object name", which is user-settable, and a "localized descriptor" which is set by the manufacturer. The initial value of the object name is all zeros, which indicates that the user has not specified a name. The processing rule is to user the object name if it has been specified, otherwise to use the localized descriptor. The following Clauses reflect this rule.

7.1. CM4-Milan mapping

This Adaptation defines a mapping between CM4 elements and Milan elements. Figure 3 is an overview diagram of this mapping for the simplest use case, in which there is one Clock Domain and one Audio Unit.

Details of the CM4 elements in Figure 3 are given in the subsequent sections.

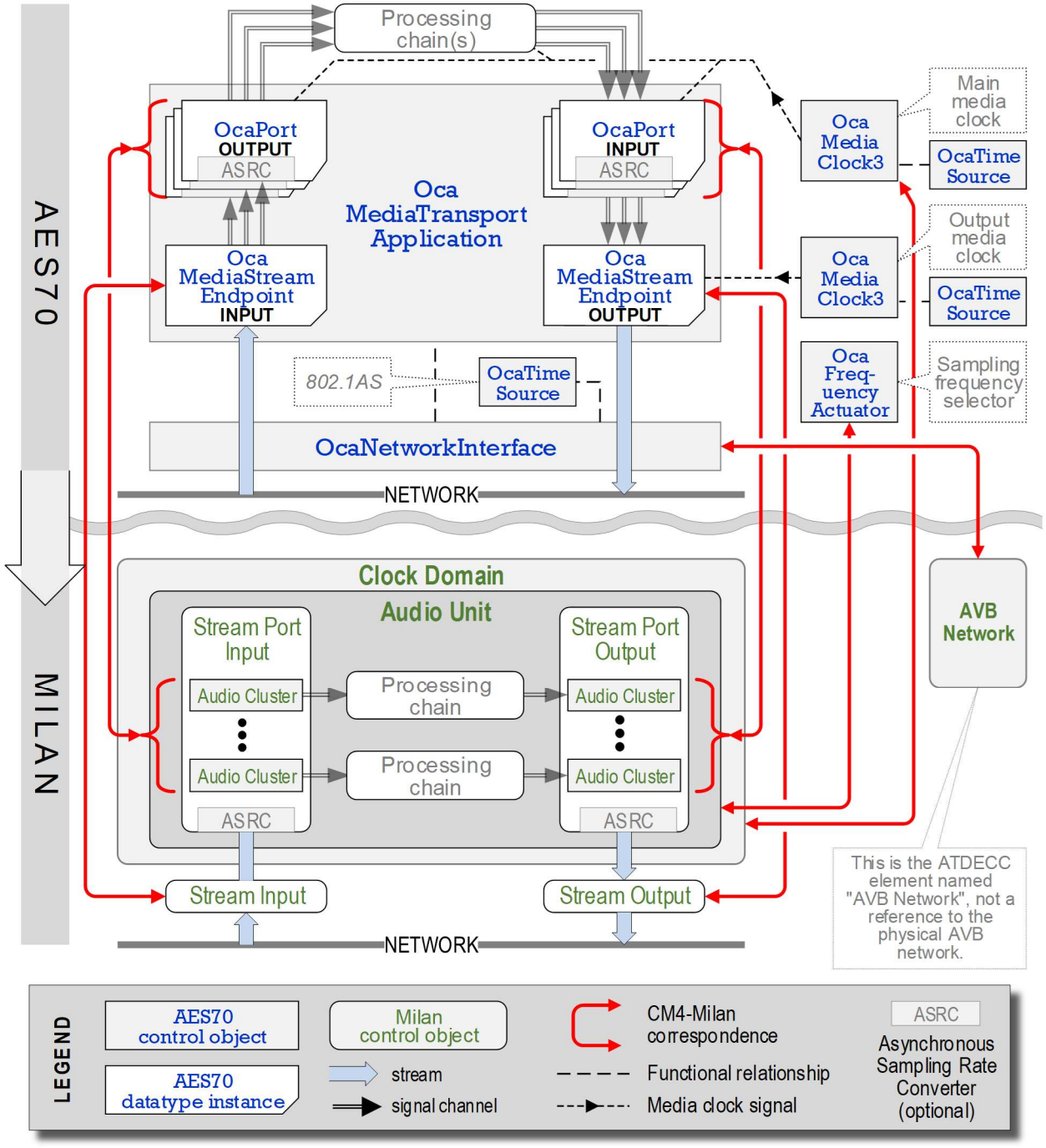

Figure 3. CM4->Milan mapping
Showing the use case with one Clock Domain and one Audio Unit.

7.2. Class OcaNetworkInterface

A Device's interface to an external physical network shall be controlled by an instance of the class OcaNetworkInterface.

For this Adaptation, each instance of the class OcaNetworkInterface shall represent one Milan AVB Interface instance. A non-redundant Device shall have one AVB Interface and one OcaNetworkInterface instance; a redundant Device shall have two of each.

Adaptation-specific property values of OcaNetworkInterface instances shall be as specified by Table 1.

Table 1. Adaptation-specific property values of class OcaNetworkInterface

Property	Value
OcaString Role	Value of the localized_description field of the AVB_INTERFACE descriptor.
OcaString Label	Value of the object_name field of the AVB_INTERFACE descriptor.
OcaUint16 GroupID	Always zero.
OcaUint16 Precedence	Value of the descriptor_index field of the AVB_INTERFACE descriptor plus one (1 for primary, 2 for secondary).
OcaAdaptationIdentifier AdaptationIdentifier	Always "OcaMilan".
OcaAdaptationData CurrentAdaptationData	See Clause 7.2.1.
OcaAdaptationData RequestedAdaptationData	Shall not be used by this Adaptation, because the value of CurrentAdaptationData is read-only.
OcaNetworkInterfaceStatus Status	See Clause 7.2.3.
OcaCounterSet CounterSet	See Clause 7.2.4.

7.2.1. Property OcaNetworkInterface.CurrentAdaptationData

CM4 defines OcaNetworkInterface.CurrentAdaptationData (datatype OcaBlob) as a flexible property whose format and contents depend on the Adaptation. For this Adaptation, the format and contents of this property shall be defined by the datatype MilanNetworkInterfaceAdaptationData, whose fields are defined in Table 2, and the value shall be read-only.

Table 2. Field values of datatype MilanNetworkInterfaceAdaptationData
(datatype of value of OcaNetworkInterface.CurrentAdaptationData)

Field	Value
OcaONo TimeSourceONo	Object number of an OcaTimeSource object representing the [IEEE-802.1AS] Time Source, as specified by Clause 7.6.1.
OcaBlobFixedLength<6> MacAddress	MAC Address associated with the AVB Interface.

7.2.2. Property OcaNetworkInterface.RequestedAdaptationData

For this Adaptation, the CurrentAdaptationData property shall be read-only. The RequestedAdaptationData property and the SetRequestedAdaptationData(...) method shall not be supported.

7.2.3. Property OcaNetworkInterface.Status

The operational status of an OcaNetworkInterface object shall be represented by its Status property, a structure of datatype OcaNetworkInterfaceStatus containing the fields listed in Table 3.

Table 3. Field values of datatype OcaNetworkInterfaceStatus
(datatype of OcaNetworkInterface.Status)

Field	Value
OcaNetworkInterfaceState State	Ready if LINK_UP counter > LINK_DOWN, NotReady otherwise
OcaAdaptationData AdaptationData	See Clause 7.2.3.1.

As shown in Table 3, the Status.State parameter is dependent on two of the counters contained in the OcaNetworkInterface.CounterSet property, namely LINK_UP and LINK_DOWN.

- When LINK_UP has a higher count than LINK_DOWN, the Status.State field shall automatically be set to Ready.
- When the LINK_DOWN count becomes higher or equal, the Status.State field shall automatically be set to NotReady.

For details on the particular counters specified by OcaNetworkInterface.CounterSet, see Clause 7.2.4.

7.2.3.1. Field Status.AdaptationData

Status.AdaptationData (datatype OcaBlob) is a flexible property whose format and contents depend on the Adaptation. For this standard, the format and contents of this property shall be have the datatype OcaList<MilanMSRPMapping> and shall represent the list of MSRP mappings in the AVB Interface, as defined by the Milan GET_AVB_INFO command. The datatype MilanMSRPMapping is defined in Table 4.

Table 4. Field values of datatype MilanMSRPMapping
(datatype of each list item in value of Status.AdaptationData)

Field	Value
OcaUint8 ClassID	Traffic class
OcaUint8 Priority	Priority of the given traffic class
OcaUint16 VLanID	VLAN ID of the given traffic class

7.2.4. Property OcaNetworkInterface.CounterSet

The property OcaNetworkInterface.CounterSet shall contain OcaCounters and corresponding OcaID16 counter identifiers, each representing the counters within the AVB Interface. These counters are listed in Table 5.

Table 5. Counters in OcaNetworkInterface.CounterSet

Counter Name	Counter ID	Requirement
LINK_UP	1	Mandatory
LINK_DOWN	2	Mandatory
FRAMES_TX	3	Optional
FRAMES_RX	4	Optional
RX_CRC_ERROR	5	Optional
GPTP_GM_CHANGED	6	Mandatory

7.2.4.1. Counter notifier requirement

At least one OcaCounterNotifier instance shall be attached to each of the mandatory counters in Table 5.

7.2.4.2. Field CounterSet.ID

For this standard, the value of the OcaNetworkInterface.CounterSet.ID field shall be the object number of the OcaNetworkInterface instance containing the Counterset.

7.2.5. Unsupported OcaNetworkInterface methods

The following OcaNetworkInterface shall not be supported by this Adaptation. These methods shall be defined in the Device API, but when called shall return the OcaStatus value NotImplemented:

- SetGroupID(...)
- SetPrecedence(...)
- SetRequestedAdaptationData(...)
- GetRequestedAdaptationData(...)
- ResetCounters(...)
- ApplyCommand(...)

7.3. Class OcaMediaTransportApplication

Central to CM4 is the class OcaMediaTransportApplication, a subclass of the NAC base class OcaNetworkApplication. A Device compliant with this Adaptation shall instantiate exactly **one** OcaMediaTransportApplication object per Entity.

Adaptation-specific property values of OcaNetworkInterface instances shall be as specified by Table 6.

Table 6. Adaptation-specific property values of OcaMediaTransportApplication

Property	Value
OcaString Role	Value of the model_name_string field of the ENTITY descriptor.
OcaString Label	Value of the entity_name field of the ENTITY descriptor.
OcaList<OcaNetworkInterface-Assignment> NetworkInterfaceAssignments	See Clause 7.3.1.
OcaAdaptationIdentifier AdaptationIdentifier	Always "OcaMilan".
OcaAdaptationData AdaptationData	See Clause 7.3.2.
OcaCounterSet CounterSet	Empty Counterset.
OcaList<OcaPort> Ports	See Clause 7.3.2.1.
OcaMap<OcaPortID, OcaPortClockMapEntry> PortClockMap	See Clause 7.3.4.
OcaUint16 MaxInputEndpoints	The fixed number of elements in the Endpoints property of type Input.
OcaUint16 MaxOutputEndpoints	The fixed number of elements in the Endpoints property of type output.
OcaUint16 MaxPortsPerChannel	Always zero.
OcaUint16 MaxChannelsPerEndpoint	Index of the highest channel format supported by any Endpoint.
OcaList <OcaMediaStreamModeCapability> MediaStreamModeCapabilities	See Clause 7.3.5.
OcaMediaTransportTimingParameters TransportTimingParameters	All fields shall be zero. Read-only.
OcaList<OcaMediaStreamEndpoint> Endpoints	See Clause 7.3.6.
OcaMap<OcaMediaStreamEndpointID, OcaMediaStreamEndpointStatus> EndpointStatuses	See Clause 7.3.7.
OcaMap<OcaID16, OcaCounterSet> EndpointCounterSets	See Clause 7.3.8.
OcaList<OcaONo> TransportSessionControlAgentONos	Object Number(s) of related MilanOcaMediaTransportSessionAgent instance(s). A Device shall implement one such instance per Entity. See Clause 7.4.

7.3.1. Property OcaMediaTransportApplication.NetworkInterfaceAssignments

The NetworkInterfaceAssignments property shall be a list of items of datatype OcaNetworkInterfaceAssignment. For this Adaptation, the field values of this datatype shall be as specified by Table 7.

In this Adaptation, the NetworkInterfaceAssignments property is read-only. Thus the SetNetworkInterfaceAssignments(...) method shall not be supported.

In a Device that does not implement redundant networking, the NetworkInterfaceAssignments list shall contain exactly **one** OcaNetworkInterfaceAssignment item, referencing the only OcaNetworkInterface object in the Device.

In a Device that implements redundant networking, the NetworkInterfaceAssignments property list shall contain exactly **two** OcaNetworkInterfaceAssignment items. The first item shall have the ID 1 and shall designate the primary OcaNetworkInterface object in the Device, while the second item shall have the ID 2 and shall designate the secondary OcaNetworkInterface object.

Table 7. Field values of datatype OcaNetworkInterfaceAssignment
(datatype of each list item in OcaMediaTransportApplication.NetworkInterfaceAssignments.)

Field	Value
OcaID16 ID	1 (for primary) or 2 (for secondary in redundant configurations).
OcaONo NetworkInterfaceONo	Object number of assigned OcaNetworkInterface object.
OcaBlob AdvertisementProtocolDescriptor	Empty blob
OcaList<OcaParameterRecord> AdvertisedServices	Empty record

7.3.2. Property OcaMediaTransportApplication.AdaptationData

CM4 defines OcaMediaTransportApplication.AdaptationData (datatype OcaBlob) as a flexible property whose format and contents depend on the Adaptation. For this standard, the format and contents of this property shall be defined by the datatype MilanMediaTransportAdaptationData as specified by Table 8.

Table 8. Field values of datatype MilanMediaTransportAdaptationData
(datatype of value of OcaMediaTransportApplication.AdaptationData)

Field	Value
OcaUint64 EntityID	Equal to the value of the entity_id field of the ENTITY descriptor of this Device.
OcaUint32 ProtocolVersion	Equal to the value of the protocol_version parameter provided by the GET_MILAN_INFO command.
OcaUint32 CertificationVersion	Equal to the value of the certification_version parameter provided by the GET_MILAN_INFO command.
OcaBoolean RedundancySupported	Equal to the state of the REDUNDANCY flag in the features_flags parameter provided by the GET_MILAN_INFO command.
OcaList<MilanAudioUnit> AudioUnits	List of descriptors of Audio Units associated with this OcaMediaTransportApplication object. See Clause 7.3.2.1.

7.3.2.1. Field ApplicationData.AudioUnits

The ApplicationData.AudioUnits field shall be a list of items of datatype MilanAudioUnit. There shall be one list entry for each Audio Unit associated with this OcaMediaTransportApplication object.

Fields of MilanAudioUnit shall be as defined in Table 9.

Table 9. Field values of datatype MilanAudioUnit
(datatype of each list item in OcaMediaTransportApplication.AdaptationData.AudioUnits)

Field	Value
OcaString Name	ATDECC name of Audio Unit. = AUDIO_UNIT.object_name
OcaONo ClockONo	ONo of OcaMediaClock3 object associated with this Audio Unit. See Clause 7.5.1.
OcaONo RateSelectorONo	ONo of OcaFrequencyActuator object that selects the clock frequency this Audio Unit is currently using.
OcaInterval<OcaUint16> InputOcaPortIndexRange	Index range of associated OCA Input Ports
OcaInterval<OcaUint16> OutputOcaPortIndexRange	Index range of associated OCA Output Ports

7.3.3. Property OcaMediaTransportApplication.Ports

The Ports property of the OcaMediaTransportApplication object shall contain exactly one OcaPort instance for each anchored Audio Cluster.

An anchored Audio Cluster is an Audio Cluster that belongs to an anchored Stream Port Input or Stream Port Output. An anchored Stream Port Input or Stream Port Output is one that belongs to an Audio Unit.

Field values of the OcaPort instances shall be as defined in Table 10.

For this Adaptation, the number of OcaPort elements in OcaMediaTransportApplication is static. Therefore, the AddPort(...) and DeletePort(...) methods shall not be supported. See Clause 7.3.9 for details on these and other interface methods in OcaMediaTransportApplication.

Table 10. Field values of datatype OcaPort
(datatype of each list item in OcaMediaTransportApplication.Ports)

Field	Value
OcaPortID ID	See subfields ID.Index and ID.Direction below
OcaUint16 ID.Index	See Clause 7.3.3.1.
OcaIODirection ID.Direction	Output if the Audio Cluster belongs to a Stream Port Input; Input if the Audio Cluster belongs to a Stream Port Output. To better understand this seeming contradiction, see the explanation of Stream Ports in Clause A.3.4.1.
OcaString Role	Equal to the object_name field of the STREAM_INPUT descriptor, if this field is nonempty; otherwise, equal to the localized_description field
OcaONo Owner	Object number of the containing OcaMediaTransportApplication object.

7.3.3.1. Field OcaPort.ID.Index

The Output OCA Port with index 1 shall reflect the first Audio Cluster of the first Stream Port Input of the first Audio Unit in the Device's Entity model. Likewise, the Input OCA Port with index 1 shall reflect the first Audio Cluster of the first Stream Port Output of the first Audio Unit.

Indexing shall continue through all the AUDIO_CLUSTER descriptors in the Audio Unit, ending with the last AUDIO_CLUSTER descriptor of the last Stream Port Input or Stream Port Output of the last Audio Unit.

7.3.4. Property OcaMediaTransportApplication.PortClockMap

For this Adaptation, entries of the property OcaMediaTransportApplication.PortClockMap shall reflect the configuration of the associated Audio Unit's Stream Port Input or Stream Port Output.

The PortClockMap shall contain exactly one entry for each OcaPort in the propertyOcaMediaTransportApplication.Ports. Each map item's key shall correspond to the ID of the referenced Ports item, while the map item's value shall be of type OcaPortClockMapEntry, specified by Table 11.

Table 11. Field values of datatype OcaPortClockMapEntry
(datatype of each map item value in OcaMediaTransportApplication.PortClockMap)

Field	Value
OcaONo ClockONo	Object number of the OcaMediaClock3 representing the Clock Domain referenced by the Audio Unit to which the OCA Port belongs. For details on the OcaMediaClock3 class see Clause 7.5.
OcaSamplingRateConverterType SRCType	Set according to the port_flags of the Stream Port Input or Stream Port Output which the OcaPort represents, as follows: • Asynchronous if the ASYNC_SAMPLE_RATE_CONV flag is 1 and the SYNC_SAMPLE_RATE_CONV flag is 0 or 1. • Synchronous if the ASYNC_SAMPLE_RATE_CONV flag is 0 and the SYNC_SAMPLE_RATE_CONV flag is 1. • None if both flags are 0, indicating the absence of a sampling rate converter.

7.3.5. Property OcaMediaTransportApplication.MediaStreamModeCapabilities

The OcaMediaTransportApplication.MediaStreamModeCapabilities property shall be a list of items of datatype OcaMediaStreamModeCapability.

MediaStreamModeCapabilities shall describe all possible stream formats supported by all Stream Inputs and Stream Outputs of the associated Entity.

This Adaptation supports the following stream formats, as specified by [AVnu-Milan(Summary of Base audio stream formats)]:

- Clock Reference Stream Format (CRF).
- AVTP Audio Stream Format (AAF).

The MediaStreamModeCapabilities list may contain OcaMediaStreamModeCapability descriptors for either of both of these stream formats.

The MediaStreamModeCapabilities list and the capabilities listed therein are static; consequently, the SetMediaStreamModeCapabilities(...) method shall not be supported. See Clause 7.3.9 for details on this and other interface methods in OcaMediaTransportApplication.

7.3.5.1. CRF formats

Table 12 specifies the fields of OcaMediaStreamModeCapability instances for CRF Streams.

Table 12. Field values of datatype OcaMediaStreamModeCapability for CRF Streams
(datatype of each list item in OcaMediaTransportApplication.MediaStreamModeCapabilities)

Field	Value
OcaID16 ID	No Milan-specific requirements
OcaString Name	Empty string
OcaMediaStreamModeCapabilityDirection Direction	• The Input bit shall be 1 if this capability applies to Input Endpoints, 0 otherwise. • The Output bit shall be 1 if this capability applies to Output Endpoints, 0 otherwise.
OcaList<OcaMediaFrameFormat> FrameFormatList	Exactly one element: CRF_MILAN
OcaList<OcaMimeType> EncodingTypeList	Empty list
OcaList<OcaFrequency> SamplingRateList	List with exactly one element, corresponding to the value defined by the base_frequency of the associated Stream Format String.
OcaList<OcaUint16> ChannelCountList	List containing exactly one element with value zero
OcaInterval<OcaUint16> ChannelCountRange	Empty range
OcaList<OcaTimeInterval> PacketTimeList	List containing exactly one element with value 125µs
OcaInterval<OcaTimeInterval> PacketTimeRange	Empty range

7.3.5.2. AAF formats

Table 13 specifies the fields of OcaMediaStreamModeCapability instances for AAF Streams..

Table 13. Field values of datatype OcaMediaStreamModeCapability for AAF Streams
(datatype of each list item in OcaMediaTransportApplication.MediaStreamModeCapabilities)

Field	Value
OcaID16 ID	No Milan-specific requirements
OcaString Name	Empty string
OcaMediaStreamModeCapabilityDirection Direction	• The Input bit shall be 1 if this capability is referenced by an Input Endpoint, 0 otherwise. • The Output bit shall be 1 if this capability is referenced by an Output Endpoint, 0 otherwise.
OcaList<OcaMediaFrameFormat> FrameFormatList	List containing exactly one element with value AAF

Field	Value
OcaList<OcaMimeType> EncodingTypeList	List containing exactly one element with value "audio/L32". See [AES70-1(MIME media types)] and AES70-2A(OcaMimeType)].
OcaList<OcaFrequency> SamplingRateList	List with exactly one element containing the nominal sample rate (nsr) of each associated Stream Format String.
OcaList<OcaUint16> ChannelCountList	• If the Stream Format String's ut bit within the associated Stream Format String is 1, this shall be an empty list. • Otherwise, this shall be a list containing exactly one element: channels_per_frame.
OcaInterval<OcaUint16> ChannelCountRange	• If the Stream Format String's ut bit within the associated Stream Format String is 1, the interval shall be [1, channels_per_frame]. • Otherwise, this shall be an empty interval. See [AES70-2A(OcaInterval)].
OcaList<OcaTimeInterval> PacketTimeList	List containing exactly one element with value 125µs
OcaInterval<OcaTimeInterval> PacketTimeRange	Empty range

7.3.6. Property OcaMediaTransportApplication.Endpoints

The property OcaMediaTransportApplication.Endpoints shall contain a list of OcaMediaStreamEndpoint instances, each representing a Stream Input or Stream Output. Therefore, the total number of Endpoints managed by the OcaMediaTransportApplication object shall be the sum of all Stream Input and Stream Output counts in the Entity. This value is static. Hence, the AddEndpoint(...) and DeleteEndpoint(...) methods shall not be supported. See Clause 7.3.9 for details on these and other interface methods in OcaMediaTransportApplication.

In a Device that implements redundant networking, there shall be two identical Endpoints, one attached to each of the OcaNetworkInterface instances that represent the two networks.

7.3.6.1. Field values of Endpoints

In this Adaptation, the field values of each Endpoint shall depend on the Endpoint's Direction.

7.3.6.1.1. Input Endpoints

Each Input Endpoint shall represent a Stream Input and shall have field values as specified by Table 14.

NOTE Table 14 mentions both OcaMediaStreamModeCapability and OcaMediaStreamMode datatypes. OcaMediaStreamModeCapability is a searchable datatype that may describe more than one stream mode capability. OcaMediaStreamMode describes one specific stream mode. See [AES70-2A] for their normative definitions.

Table 14. Field values of datatype OcaMediaStreamEndpoint for Input Endpoints
(datatype of each list item in OcaMediaTransportApplication.Endpoints)

Field	Value
OcaMediaStreamEndpointID IDInternal	STREAM_INPUT descriptor's index plus 1.
OcaBlob IDExternal	See Clause 7.3.6.2.
OcaIODirection Direction	Input
OcaString UserLabel	Equal to the object_name field of the STREAM_INPUT descriptor, if this field is nonempty; otherwise, equal to the localized_description field
OcaList<OcaID16> NetworkAssignmentIDs	See Clause 7.3.6.3.
OcaList<OcaID16> StreamModeCapabilityIDs	List of IDs of OcaMediaStreamModeCapability instances that reference the capabilities enumerated in the OcaMediaTransportApplication.MediaStreamModeCapabilities property - see Clause 7.3.5. The list shall reflect the stream mode capabilities of this endpoint.
OcaONo ClockONo	Zero, indicating the absence of an associated OcaMediaClock3 object. Milan Stream Inputs are clocked by the streams they receive, so they do not use clocking from the Device.
OcaBoolean ChannelMapDynamic	TRUE for Input Endpoints. FALSE for Endpoints using the CRF format. See Clause 7.3.6.4.
OcaMultiMap<OcaUint16, OcaPortID> ChannelMap	Multi-map whose keys shall be the indices of the Endpoint's channels (one-based), and values shall be the IDs of the OcaMediaTransportApplication's Output ports. See Clause 7.3.6.4.
OcaMediaStreamMode CurrentStreamMode	OcaMediaStreamMode instance that shall specify the stream format described by the Stream Format String residing in the current_format field of the STREAM_INPUT descriptor.
OcaSecurityType SecurityType	Always None.
OcaMediaStreamCastMode StreamCastMode	Always Multicast.
OcaAdaptationData AdaptationData	See Clause 7.3.6.5.
OcaID16 RedundantSetID	Zero for Endpoints in non-redundant Devices. For Endpoints in redundant Devices, value shall be equal to the IDInternal value of the Endpoint associated with the primary OcaNetworkInterface object.

7.3.6.1.2. Output Endpoints

Output Endpoints shall represent Stream Outputs, and shall contain the same field values as specified for Input Endpoints in Table 14, except with STREAM_INPUT replaced by STREAM_OUTPUT, and with exceptions as listed in Table 15.

Table 15. Field values of datatype OcaMediaStreamEndpoint for Output Endpoints
(datatype of each list item in OcaMediaTransportApplication.Endpoints)

Field	Value
OcaMediaStreamEndpointID IDInternal	The STREAM_OUTPUT descriptor's index plus 1001. NOTE In AES70, all Endpoints (whether input or output) share a common ID pool. Adding an offset for the Output Endpoints ensures that there are no conflicts with IDs of the Input Endpoints.
OcaBlob IDExternal	See Clause 7.3.6.2.
OcaIODirection Direction	Output
OcaList<OcaID16> NetworkAssigmentIDs	See Clause 7.3.6.3.
OcaONo ClockONo	ONo of the OcaMediaClock3 object that corresponds to the Clock Domain of the associated Stream Output. See Clause 7.5.1.1.
OcaBoolean ChannelMapDynamic	For AAF Endpoints: • FALSE if the STREAM_PORT_OUTPUT has an AUDIO_MAP descriptor defined; TRUE otherwise. • Value shall be identical for all Output AAF Endpoints in the Device. For CRF Endpoints, always FALSE. See Clause 7.3.6.4.
OcaMultiMap<OcaUint16, OcaPortID> ChannelMap	Map whose keys are the indices of the Endpoint's channels (one-based), and whose values are the IDs of the associated OcaMediaTransportApplication's Input Ports. See Clause 7.3.6.4.
OcaAdaptationData AdaptationData	See Clause 7.3.6.5.

7.3.6.2. Endpoint field IDExternal

The IDExternal field shall uniquely identify an Endpoint across the network. When establishing a connection between two Devices, the IDExternal field of the Output Endpoint in the remote Device shall be used to configure the local Session, as specified by Clause 7.4.1.1.

IDExternal (datatype OcaBlob) is a flexible property whose format and contents depend on the Adaptation. For this standard, the format and contents of this property shall be as defined by the datatype MilanMediaStreamEndpointIDExternal, specified by Table 16.

Table 16. Field values of datatype MilanMediaStreamEndpointIDExternal
(datatype of value of OcaMediaStreamEndpoint.IDExternal field)

Field	Description
OcaUint64 EntityID	EntityID of this Device, as defined by entity_id field of the ENTITY descriptor.
OcaUint16 StreamIndex	Index of the STREAM_INPUT or STREAM_OUTPUT descriptor.

7.3.6.3. Endpoint field NetworkAssignmentIDs

In Milan, every STREAM_INPUT or STREAM_OUTPUT references one AVB_INTERFACE; similarly, in this Adaptation every Endpoint shall reference an OcaNetworkInterface object. To accomplish this, each Endpoint's NetworkAssignmentIDs field shall be a list containing exactly one element with values as follows:

- In the case of a non-redundant Device, the value shall be 1.
- In the case of a redundant Device, the value shall be 1 or 2, depending on whether this Endpoint is assigned to the primary (ID=1) or secondary (ID=2) OcaNetworkInterface object.

For details on network assignment IDs, see Clause 7.3.1.

7.3.6.4. Endpoint field ChannelMap

In general, a channel mapping is a data structure that specifies the routing of a stream channel to an internal signal path.

- In the Milan Entity model, a collection of channel mappings resides in each STREAM_PORT_INPUT or STREAM_PORT_OUTPUT.
- In the CM4 model, the collection of channel mappings shall reside in the field OcaMediaStreamEndpoint.ChannelMap.

A Device shall maintain ChannelMap to reflect the current channel mapping at all times, regardless of whether the mapping is statically stored in an Audio Map or dynamically maintained via a Milan control API. See Clause A.3.5 for a complete explanation of channel mapping.

7.3.6.4.1. ChannelMap contents for AAF Endpoints

The keys of ChannelMap items shall represent Stream channel IDs. The values of these IDs must be integers within the Endpoint's current channel capacity, as given by the Endpoint's CurrentStreamMode field. In this Adaptation, channel ID values shall start at 1.

The values of ChannelMap items shall values of OCA Port IDs in the OcaMediaTransportApplication object.

The ChannelMap of an Input Endpoint shall only specify Output OCA Ports; it may specify any number of them per input Stream channel. The ChannelMap of an Output Endpoint shall only specify Input OCA Ports; it may specify at most one Input OCA Port per output Stream channel.

Whether the Device supports dynamic mappings or static mappings provided by the AUDIO_MAP descriptor shall be indicated by the value of Endpoint's **ChannelMapDynamic** field.

7.3.6.4.2. **ChannelMap** contents for CRF Endpoints

A CRF Endpoint's **ChannelMap** shall be empty and the value of its **ChannelMapDynamic** field shall be **FALSE**.

7.3.6.5. Endpoint field **AdaptationData**

AdaptationData (datatype **OcaAdaptationData)** is a flexible property whose format and contents depend on the Adaptation. For this standard, the format and contents of this property shall be as defined by the datatype **MilanMediaStreamEndpointAdaptationData**, as specified by Table 17.

The individual field values in the **AdaptationData** structure depend on the **State** field of the corresponding Endpoint. See Clause 7.3.7 for details on the Endpoint states supported by this Adaptation.

In the case of Input Endpoints, the entire **AdaptationData** property shall be read-only, and thus not available for changing via the **SetEndpointAdaptationData(...)** method.

In the case of Output Endpoints, most of the fields defined in the **OcaMediaStreamEndpoint.AdaptationData** structure shall be read-only, except where otherwise specified. When calling the method **SetEndpointAdaptationData(...)**, all read-only fields of the provided **AdaptationData** structure shall have the value 0, otherwise the status **ParameterError** shall be returned.

> NOTE The purpose of this rule is both to acknowledge that no change of the read-only parameters is intended, and to allow changing other (writeable) parameters within the structure without having to query the current **AdaptationData** structure first.

For Endpoints which are in the **NotReady** state, all fields defined in the **OcaMediaStreamEndpoint.AdaptationData** structure shall be zero (or *"00:00:00:00:00:00"* in the case of the **MacAddress** field). For Endpoints which are in the **Running** or **Ready** states, field values shall be as defined in Table 17.

Table 17. Field values of datatype Milan MediaStreamEndpoint.AdaptationData**
(datatype of value of field **OcaMediaStreamEndpoint.AdaptationData**)**

Field	Value
OcaUint64 StreamID	Equal to the value of the Stream ID field of the STREAM_INPUT / STREAM_OUTPUT descriptor of the associated ENTITY.
OcaBlobFixedLength<6> MacAddress	Equal to the Stream Destination MAC Address field of the STREAM_INPUT / STREAM_OUTPUT descriptor.
OcaUint16 VlanID	Equal to the Stream VLAN ID field of the STREAM_INPUT / STREAM_OUTPUT descriptor. This field shall be writeable for Output Endpoints which are in the **NotReady** state using the method **SetEndpointAdatpationData(...)**. In all other cases, this field shall be read-only.

Field	Value
OcaUint32 BufferLength	Equal to the buffer_length field of the STREAM_INPUT / STREAM_OUTPUT descriptor.
OcaUint32 PresentationTimeOffset	For an Output Endpoint, equal to the value of the msrp_accumulated_latency field of the GET_STREAM_INFO command for the Endpoint's output Stream. For an Input Endpoints, zero. This field shall be writeable for Output Endpoints which are in the NotReady state using the method SetEndpointAdaptationData(...). In all other cases, this field shall be read-only.

7.3.7. Property OcaMediaTransportApplication.EndpointStatuses

The operational status of each Endpoint shall be represented by an item in the EndpointStatuses property, a list of items of datatype OcaMediaStreamEndpointStatus. See Table 18.

Table 18. Field values of datatype OcaMediaStreamEndpointStatus
(datatype of each list item in OcaMediaTransportApplication.EndpointStatuses)

Field	Value
OcaMediaStreamEndpointState State	Input, see Clause 7.3.7.1. Output, see Clause 7.3.7.2.
OcaUint16 ErrorCode	Always zero

7.3.7.1. Field State for Input Endpoints

Value of the State field for an Input Endpoint shall be as follows:

- If the value of the STREAM_DEST_MAC_VALID flag of the associated Stream Input is FALSE, then the value of State shall be NotReady.
- Otherwise, if the value of the MEDIA_LOCKED counter of the Stream Input exceeds the value of its MEDIA_UNLOCKED counter, then the value of State shall be Running.
- In all other cases, the value of State shall be Ready.

7.3.7.2. Field State for Output Endpoints

Value of the State field for an Output Endpoint shall be as follows:

- If the value of the STREAM_DEST_MAC_VALID flag of the associated Stream Output is FALSE, then the value of State shall be NotReady.
- Otherwise, if the value of the STREAM_START counter of the Stream Output is exceeds the value of its STREAM_STOP counter, then the value of State shall be Running.
- In all other cases, the value of State shall be Ready.

7.3.8. Property OcaMediaTransportApplication.EndpointCounterSets

The OcaMediaTransportApplication.EndpointCounterSets property shall contain one OcaCounterSet per Endpoint. The counters inside each of these Countersets shall depend on the Direction parameter of each corresponding OcaMediaStreamEndpoint.

7.3.8.1. Counter IDs for Input Endpoints

Countersets belonging to Input Endpoints shall contain OcaCounters that represent the Milan counters within the Stream Input. These OcaCounters and their corresponding OcaID16 counter identifiers are listed in Table 19.

Table 19. Counter IDs for Input Endpoints in OcaMediaTransportApplication.EndpointCounterSets

Counter Name	Counter ID	Requirement
MEDIA_LOCKED	1	Mandatory
MEDIA_UNLOCKED	2	Mandatory
STREAM_INTERRUPTED	3	Mandatory
SEQ_NUM_MISMATCH	4	Mandatory
MEDIA_RESET	5	Mandatory
TIMESTAMP_UNCERTAIN	6	Mandatory
UNSUPPORTED_FORMAT	9	Mandatory
LATE_TIMESTAMP	10	Mandatory
EARLY_TIMESTAMP	11	Mandatory
FRAMES_RX	12	Mandatory

7.3.8.2. Counter IDs for Output Endpoints

Countersets belonging to Output Endpoints shall contain OcaCounters that represent the Milan counters within the Stream Output. These OcaCounters and their corresponding OcaID16 counter identifiers are listed in Table 20.

Table 20. Counter IDs for Output Endpoints in OcaMediaTransportApplication.EndpointCounterSets

Counter Name	Counter ID	Requirement
STREAM_START	1	Mandatory
STREAM_STOP	2	Mandatory
MEDIA_RESET	3	Mandatory
TIMESTAMP_UNCERTAIN	4	Mandatory

Counter Name	Counter ID	Requirement
FRAMES_TX	5	Mandatory

7.3.8.3. Field EndpointCounterSet.ID

OcaCounterSet.ID (datatype OcaBlob) is a flexible property whose format and contents depend on the Adaptation. For this standard, the format and contents of this property shall be as defined by the datatype MilanEndpointCounterSetID, defined in Table 21.

Table 21. Field values of datatype MilanEndpointCounterSetID
(datatype of value of field EndpointCounterSet.ID)

Field	Value
OcaONo OwnerONo	ONo of the owning OcaMediaTransportApplication object.
OcaPropertyID CounterSetsPropertyID	PropertyID of the EndpointCounterSets property of the OcaMediaTransportApplication class.
OcaMediaStreamEndpointID EndpointID	ID of the Endpoint containing the Counterset.

7.3.9. OcaMediaTransportApplication methods requiring special consideration

Table 22 lists methods of the class OcaMediaTransportApplication for which there are special considerations to be taken in this Adaptation.

Table 22. OcaMediaTransportApplication methods requiring special consideration

Method	Consideration
SetEndpointMediaStreamMode(...)	Shall only be callable if the Endpoint is in the NotReady state. In all other cases, this method shall return the status InvalidRequest.
SetEndpointChannelMap(...)	Shall only be callable if the ChannelMapDynamic flag is TRUE, otherwise this method shall return the status InvalidRequest. In the case of Output Endpoints, mapping changes shall only be allowed when the Endpoint's state is NotReady.
SetEndpointAdaptationData(...)	Shall only be callable for Output Endpoints which are in the NotReady state. In all other cases, this method shall return the status InvalidRequest. Only the VlanID and PresentationTimeOffset fields of the MilanOcaMediaStreamEndpointAdaptationData datatype shall be modifiable. See Clause 7.3.6.5 for details.

Method	Consideration
SetNetworkInterfaceAssignments(...) SetAdaptationData(...) ResetCounters(...) AddPort DeletePort(...) SetPortClockMap(...) SetPortClockMapEntry(...) DeletePortClockMapEntry(...) SetTransportTimingParameters(...) AddEndpoint(...) DeleteEndpoint(...) ApplyEndpointCommand(...) ResetEndpointCounterSet(...) SetMediaStreamModeCapabilities(...) SetTransportSessionControlAgentONos(...)	Shall not be supported by this Adaptation. These methods shall be defined in the Device API, but when called shall return the OcaStatus value NotImplemented.

7.4. Class MilanOcaMediaTransportSessionAgent

This Adaptation uses CM4's Session (Definition 20) mechanism to model the behavior of the Session State Machine responsible for establishing Stream connections.

The binding, reserving and connecting of Input Endpoints to AVTP Streams shall be managed by instance(s) of the class MilanOcaMediaTransportSessionAgent, this Adaptation's subclass of the Core AES70 class OcaMediaTransportSessionAgent.

A compliant Device shall instantiate exactly one MilanOcaMediaTransportSessionAgent object for every Entity. Adaptation-specific property values for this object shall be as specified by Table 23.

Table 23. Adaptation-specific property values of class OcaMediaTransportSessionAgent

Property	Value
OcaString SessionType	"OcaMilan".
OcaList<OcaMediaTransportSession> Sessions	list with one entry for each Input Endpoint in the associated OcaMediaTransportApplication object. See Clause 7.4.1.
OcaMap<OcaMediaTransportSessionID, OcaMediaTransportSessionStatus> SessionStatuses	See Clause 7.4.2.
OcaAdaptationData AdaptationData	Empty blob.

7.4.1. Property MilanOcaMediaTransportSessionAgent.Sessions

The property MilanOcaMediaTransportSessionAgent.Sessions shall contain a list of ***Session descriptors***. Each Session descriptor shall be an instance of the MilanOcaMediaTransportSession datatype. Sessions shall contain exactly one Session descriptor for each Input Endpoint in the OcaMediaTransportApplication object.

MilanOcaMediaTransportSession field values are specified by Table 24.

The format of Sessions property Adaptation-specific parameter values for individual MilanOcaMediaTransportSession elements shall be as specified by Table 24.

Table 24. Field values of datatype MilanOcaMediaTransportSession
(datatype of MilanOcaMediaTransportSessionAgent.Sessions)

Field	Value
OcaMediaTransportSessionID IDInternal	Shall follow the same numbering scheme as the IDInternal field of the associated Input Endpoint in the OcaMediaTransportApplication object, as specified by Clause 7.3.6.1.
OcaBlob IDExternal	Empty blob.
OcaString UserLabel	Empty string.
OcaBoolean StreamingEnabled	TRUE if the BOUND flag is 1 **and** the STREAMING_WAIT flag is 0, as returned by the GET_STREAM_INFO command. FALSE otherwise.
OcaAdaptationData AdaptationData	Empty blob.
OcaList< *OcaMediaTransportSessionConnection>* Connections	One Connection descriptor. See Clause 7.4.1.1.
OcaMap< *OcaMediaTransportSessionConnectionID,* *OcaMediaTransportSessionConnectionState>* ConnectionStates	Map containing exactly one instance of the OcaMediaTransportSessionConnectionState datatype. Map key shall be the ID of the single Connection descriptor in the Connections property. In the map item, the LocalEndpointState field shall be derived from the state of the local Input Endpoint corresponding to this Session, following the formula specified by Clause 7.3.7.1. Value of the RemoteEndpointState field shall always be Unknown.

7.4.1.1. Field MilanOcaMediaTransportSession.Connections

The field MilanOcaMediaTransportSession.Connections shall contain a list of ***Connection descriptors.*** Each Connection descriptor shall be an instance of the OcaMediaTransportSessionConnection datatype.

Each session descriptor shall contain exactly one Connection, where the LocalEndpointID is the IDInternal of the Endpoint it represents, which shall also be identical to the IDInternal of the Session itself. Sessions and Connections shall be static for a given configuration, and thus the AddSession(...) and DeleteSession(...) methods shall not be supported.

The Adaptation-specific field values for individual OcaMediaTransportSessionConnection elements shall be as specified by Table 25.

Table 25. Field values of datatype OcaMediaTransportSessionConnection
(datatype of each list item in MilanOcaMediaTransportSession.Connections)

Field	Value
OcaMediaTransportSession-ConnectionID ID	1
OcaMediaStreamEndpointID LocalEndpointID	Equal to the value of the IDInternal field of the Input Endpoint in the local OcaMediaTransportApplication object.
OcaBlob RemoteEndpointID	Equal to the value of the IDExternal of the Output Endpoint in the remote OcaMediaTransportApplication object. If the Input Endpoint is not bound, all fields shall be 0. See Clause 7.3.6.2.

7.4.2. Property MilanOcaMediaTransportSessionAgent.SessionStatuses

The operational status of each OcaMediaTransportSession element shall be represented by the SessionStatuses property. This property shall contain one element for each Session in the MilanOcaMediaTransportSessionAgent, and therefore one element for each corresponding Input Endpoint in the OcaMediaTransportApplication object - see Clause 7.3.6.

The Adaptation-specific field values for each OcaMediaTransportSessionStatus structure shall be as listed in Table 26.

Table 26. Field values of datatype OcaMediaTransportSession.SessionStatuses
(datatype of each list item in MilanOcaMediaTransportSessionAgent.SessionStatuses)

Field	Value
OcaMediaTransportSessionState State	See Clause 7.4.2.1.
OcaAdaptationData AdaptationData	See Clause 7.4.2.2.

7.4.2.1. Field SessionStatus.State

In this Adaptation, a Session's State property shall be used to represent the Input Endpoint's state. Table 27 maps the possible states of a STREAM_INPUT to the OcaMediaTransportSessionState of the Session which represents it.

Only the Session states Unconfigured, Configured, and ConnectedStreaming shall have any significance in this Adaptation. All other states shall not be used. Table 27 illustrates possible changes between states of individual Sessions.

Table 27. Session states

State	Description
Unconfigured	This state shall be used when the BOUND flag of the associated STREAM_INPUT is 0, as provided by the GET_STREAM_INFO command. This Session state shall be equivalent to the UNBOUND STREAM_INPUT state.
Configured	This state shall be used when the STREAM_DEST_MAC_VALID flag of the associated STREAM_INPUT is 0, or when both the STREAM-_DEST_MAC_VALID and REGISTERING_FAILED flags are 1, as provided by the GET_STREAM_INFO command. This Session state shall be equivalent to the following STREAM_INPUT states: PRB_W_AVAIL, PRB_W_DELAY, PRB_W_RESP, PRB_W_RESP2, PRB_W_RETRY, SETTLED_NO_RSV
ConnectedStreaming	This state shall be used when the STREAM_DEST_MAC_VALID flag of the associated STREAM_INPUT is 1, and the REGISTERING_FAILED flag is 0, as provided by the GET_STREAM_INFO command. This Session state shall be equivalent to the SETTLED_RSV_OK STREAM_INPUT state.

7.4.2.2. Field SessionStatus.AdaptationData

SessionStatus.AdaptationData (datatype OcaBlob) is a flexible field whose format and contents depend on the Adaptation. For this Adaptation, the format and contents of this property shall be defined by the datatype MilanSessionStatusAdaptationData, whose fields are defined in Table 28.

Table 28. Field values of datatype MilanSessionStatusAdaptationData
(datatype of value of OcaMediaTransportSessionStatus.AdaptationData)

Field	Value
MilanOcaSessionConfigSubstate Substate	Substate of a configured Session as defined in Table 29. This field shall only be defined when the Session's state is Configured.
OcaUint8 AcmpFailureCode	ACMP status reported by the last PROBE_TX_RESPONSE received. This field shall only be defined when the Session's state is Configured and the Session's substate is ProbingSource.

Field	Value
OcaUint64 SrpFailureBridgeID	Equal to the value of the msrp_failure_bridge_id parameter by the GET_STREAM_INFO command. This field shall only be defined when the Session's state is Configured and the Session's substate is ReservationError.
OcaUint8 SrpFailureCode	Equal to the value of the msrp_failure_code field returned by the GET_STREAM_INFO command. This parameter shall only be defined when the Session's state is Configured and the Session's substate is ReservationError.
OcaUint32 MSRPAccumulatedLatency	Equal to the value of the msrp_ accumulated_latency parameter returned by the GET_STREAM_INFO command. This parameter shall only be defined when the Session's state is ConnectedStreaming.

In case that a Session is in Configured state, more detailed information about the state of the input Stream being represented shall be provided by the Substate field, of datatype MilanOcaSessionConfigSubstate as defined in Table 29.

Table 29. Field values of datatype MilanOcaSessionConfigSubstate

Name	Value	Description
Undefined	0	The corresponding OcaMediaTransportSession is not in the Configured state.
SourceNotPresent	1	Awaiting ADP discovery message from the Talker. This substate is equivalent to the input Stream state PRB_W_AVAIL.
ProbingSource	2	In the process of probing the source. This substate is equivalent to the input Stream states PRB_W_DELAY, PRB_W_RESP, PRB_W_RESP2, and PRB_W_RETRY. The AcmpFailureCode field shall indicate the result of the last probe. This field is expected to be 0 the first time the source is probed.
ReservationError	3	SRP reservation failed (registering an MSRP Talker Failed attribute). When in this substate, the SrpFailureCode and SrpFailureBridgeID fields shall provide further details. This substate is equivalent to the input Stream state SETTLED_NO_RSV.

7.4.2.3. Session State Machine

Figure 4 illustrates how the state of an individual Session can change as a result of changes in the conditions defined in Table 29.

For example, the change from Configured to ConnectedStreaming happens automatically when the Talker's Stream becomes available, and changes back to Configured when the Talker's Stream becomes unavailable.

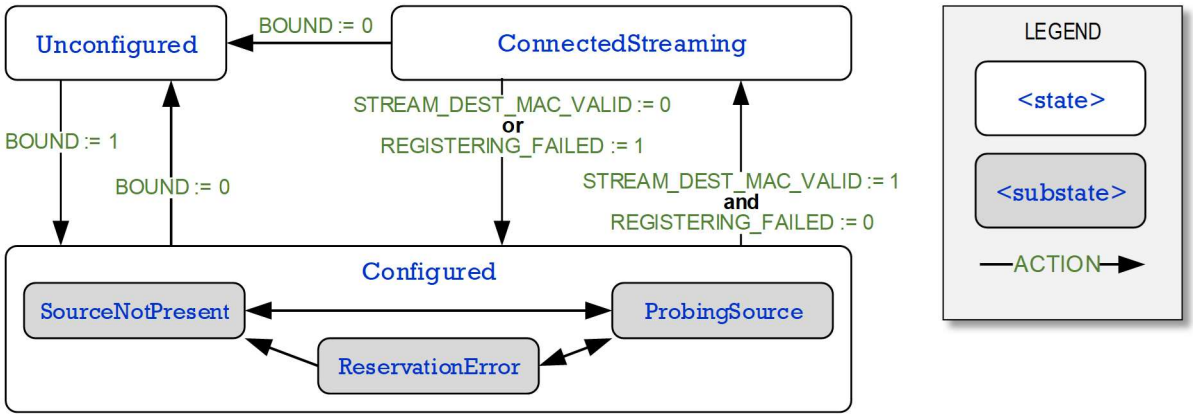

Figure 4. Session states

7.4.3. Unsupported OcaMediaTransportSessionAgent methods

MilanOcaMediaTransportSessionAgent shall not support the following interface methods inherited from the class OcaMediaTransportSessionAgent. These methods shall be defined in the Device API, but when called shall return the OcaStatus value NotImplemented:

AddSession(...)
ConfigureSession(...)
DeleteSession(...)
StartStreaming(...)
StopStreaming(...)
AddConnection(...)
DeleteConnection(...)
DeleteConnections(...)
GetAdaptationParameters(...)
SetAdaptationParameters(...)
GetSessionRegistrationAgentONo(...)
SetSessionRegistrationAgentONo(...)

7.4.3.1. Method MilanOcaMediaTransportSessionAgent.ConfigureConnection(...)

This method shall model the function of the Milan BIND_RX_COMMAND with the streaming_wait flag set implicitly.

In the call ConfigureConnection(LocalEndpointID, RemoteEndpointID):

- The LocalEndpointID parameter shall contain the IDInternal of the Input Endpoint in the local OcaMediaTransportApplication object that is being bound.
- The RemoteEndpointID parameter shall contain the IDExternal of the Output Endpoint in the remote OcaMediaTransportApplication object.

A successful ConfigureConnection(...) call shall cause the Stream Input to enter the BOUND state. Once the Stream Input is in the BOUND state, the Session's state shall become Configured as specified by Clause 7.3.2.1. See state diagram in Clause 7.3.2.3 for details.

Following the ConfigureConnection(...) call, a Controller can start the Connection's Stream flowing by calling the SetStreamingEnabled(...) method of the local OcaMediaTransportApplication object.

7.4.3.2. Method MilanOcaMediaTransportSessionAgent.ResetSession (...)

This method shall model the function of the Milan UNBIND_RX_COMMAND.

A successful ResetSession(...) call shall cause the Stream Input to enter the UNBOUND state. Once the Stream is in the UNBOUND state, the Session's Connection shall have its RemoteEndpointID set to 0, and the Session's state shall become Unconfigured as specified by Clause 7.4.2.1. See state diagram in Clause 7.4.2.3 for details.

7.4.3.3. Method MilanOcaMediaTransportSessionAgent.SetStreamingEnabled (...)

This method shall enable or disable Stream data flow.

This method shall be available for Sessions in the Configured or in the ConnectedStreaming states. If commands are successful, it will result in Session state changes as defined in Clause 7.4.2.1.

- Calling this method with the parameter value TRUE shall model the START_STREAMING command for the associated Stream.
- Calling this method with the parameter value FALSE shall model the STOP_STREAMING command for the associated Stream.

See state diagram in Clause 7.4.2.3 for details.

7.5. Class OcaMediaClock3

This Adaptation shall use the Core AES70 class OcaMediaClock3 to model the Milan Clock Domain, and also to manage the sampling rate selection of the Milan Audio Unit.

7.5.1. Use of OcaMediaClock3 to model the Clock Domain

For each Clock Domain referenced by an Audio Unit, a Device shall use one instance of OcaMediaClock3.

Each such OcaMediaClock3 object shall be listed in the PortClockMap property of the OcaMediaTransportNetworkApplication object, as specified by Clause 7.3.4.

Adaptation-specific property values of these OcaMediaClock3 objects shall be as specified by Table 30.

Table 30. Adaptation-specific property values of OcaMediaClock3

Property	Value
OcaString Role	Equal to the localized_description field of the CLOCK_DOMAIN descriptor.
OcaString Label	Equal to the object_name field of the CLOCK_DOMAIN descriptor.
OcaMediaClockAvailability Availability	Available if the value of the LOCKED counter of the associated CLOCK_DOMAIN descriptor is higher than the value of the UNLOCKED counter; Unavailable otherwise. Property is read-only.
OcaONo TimeSourceONo	ONo of the OcaTimeSource object representing the currently selected CLOCK_SOURCE in the associated CLOCK_DOMAIN descriptor.
OcaMediaClockRate CurrentRate.NominalRate	Zero. See Clause 7.5.1.1. Values of other fields of CurrentRate are undefined and shall be ignored.
OcaMap<OcaONo, OcaList<OcaMediaClockRate>> SupportedRates	When the CLOCK_DOMAIN is referenced by an Audio Unit, value shall be a map as follows: • Each key shall be the ONo of a OcaTimeSource objects which the Device uses to model the available Clock Sources, as specified by Clause 7.6.2. • Every entry shall be a list of the rates supported by the associated Audio Unit. Undefined otherwise.

7.5.1.1. Class OcaFrequencyActuator

Milan Devices all support the 48kHz sampling rate, and optionally support 96kHz and 192kHz as well. A given Clock Domain can support all of these rates. The rate currently being used is specified by the current_sampling_rate field of the AUDIO_UNIT descriptor.

For this Adaptation, an OcaFrequencyActuator object (the *rate selector*) shall be associated with each Audio Unit. The setpoint value of the rate selector shall be linked to the value of current_sampling_rate of the Audio Unit.

The ONo of the rate selector object shall be stored in the associated MilanAudioUnit descriptor stored on OcaMediaStreamEndpoint.AdaptationData.AudioUnits. See Clause 7.3.2.1.

Whenever a rate selector is used to specify the current clock rate, the value of the NominalRate field of the CurrentRate property of the associated OcaMediaClock3 object shall be zero.

7.5.2. Use of OcaMediaClock3 to model clocking of Stream Outputs

In some clocking scenarios (see Clause 7.7), a Stream Output is equipped with an Asynchronous Sample Rate Converter (ASRC) in order to output a stream with a different Clock Domain from that of the Audio

Unit originating the stream. In such cases, a separate OcaMediaClock3 object shall be created to represent the output Clock Domain, and the ONo of this object shall be stored in the associated OcaMediaStreamEndpoint's ClockONo property.

The value of the NominalRate field of the CurrentRate property of this OcaMediaClock3 object shall be zero, because the actual clock rate is determined by the format of the outgoing stream, as specified in the CurrentStreamMode field of the associated OcaMediaStreamEndpoint instance.

7.5.3. OcaMediaClock3 methods requiring special consideration

Table 31 lists interface methods of the class OcaMediaClock3 for which there are special considerations to be taken in this Adaptation.

Table 31. OcaMediaClock3 methods requiring special consideration

Method	Consideration
SetAvailability(...)	Shall not be supported by this Adaptation. This method shall be defined in the Device API, but when called shall return the OcaStatus value NotImplemented.
SetCurrentRate(...)	This method shall have two effects: • By changing the MediaClockRate parameter, it shall model the SET_SAMPLING_RATE command available to every associated Audio Unit. • By changing the TimeSourceOno parameter, it shall model the SET_CLOCK_SOURCE command available to every associated Clock Domain.

7.6. Class OcaTimeSource

This Adaptation shall use the Core AES70 class OcaTimeSource for two separate purposes. The first, specified by Clause 7.6.1, is to model the IEEE-802.1AS-related parameters of the AVB Interface. The second, specified by Clause 7.6.2, is to model the Milan Clock Source.

7.6.1. Use of OcaTimeSource to model IEEE-802.1AS time sources

A Device shall use one instance of the Core AES70 class OcaTimeSource for each AVB Interface in a Device's Entity model. The TimeDeliveryParameters property of this instance shall be used to model the parameters of the AVB Interface's time receiver, such as AVBInfo, ASPath and MilanInfo.

Adaptation-specific property values for these OcaTimeSource objects shall be as specified by Table 32.

Table 32. Adaptation-specific property values of OcaTimeSource

Property	Value
OcaTimeDeliveryMechanism TimeDeliveryMechanism	IEEE8021AS.
OcaParameterRecord TimeDeliveryParameters	See Clause 7.6.1.1.

7.6.1.1. Property OcaTimeSource TimeDeliveryParameters

The TimeDeliveryParameters Property of OcaTimeSource objects used to model IEEE-802.1AS shall contain an OcaParameterRecord with elements as defined in Table 33.

Table 33. Elements of OcaTimeSource.TimeDeliveryParameters for modeling IEEE-802.1AS

Key	Value
ieee802AsV2ParentDSParentClockIdentity	Shall reflect the gptp_grandmaster_id parameter provided by the GET_AVB_INFO command.
ieee802AsV2PortDSNeighborPropDelay	Shall reflect the propagation_delay parameter provided by the GET_AVB_INFO command.
ieee802AsV2DefaultDSDomainNumber	Shall reflect the gptp_domain_number parameter provided by the GET_AVB_INFO command.
ieee802AsV2PortDSAsCapable	Shall reflect the status of the AS_CAPABLE flag in the flags parameter provided by the GET_AVB_INFO command.
ieee802AsV2PathTraceDSTable	Shall reflect the path_sequence parameter provided by the GET_AS_PATH command.

7.6.2. Use of OcaTimeSource to model the Clock Source (MediaClocking)

A Device shall use one instance of the Core AES70 class OcaTimeSource for each Clock Source in the Device.

Clock Sources may be of type INTERNAL, EXTERNAL, or INPUT_STREAM. Required Adaptation-specific property values for OcaTimeSource objects used in these three separate cases are specified by the following clauses.

7.6.2.1. Internal Clock Sources

Adaptation-specific property values for OcaTimeSource objects used to model a internal Clock Sources shall be as specified by Table 34.

Table 34. Adaptation-specific property values of OcaTimeSource for internal clocks

Property	Description
OcaString Role	Shall reflect the localized_description field of the CLOCK_SOURCE descriptor.
OcaString Label	Shall reflect the object_name field of the CLOCK_SOURCE descriptor.
OcaTimeSourceAvailability Availability	No Milan-specific requirements.
OcaTimeDeliveryMechanism TimeDeliveryMechanism	Shall always be None.
OcaSDPString ReferenceSDPDescription	No Milan-specific requirements.
OcaString ReferenceID	No Milan-specific requirements.
OcaTimeSourceSyncStatus SyncStatus	No Milan-specific requirements.
OcaParameterRecord TimeDeliveryParameters	Shall be an empty record.

7.6.2.2. External Clock Sources

The OcaTimeSource objects used to represent external Clock Sources shall be defined at the manufacturer's discretion. The value of the object's Role property should mirror Clock Source's localized name.

7.6.2.3. Input Stream Clock Sources

Adaptation-specific property values for OcaTimeSource objects used to model Stream-based Clock Sources shall be as specified by Table 35.

Table 35. Adaptation-specific property values of OcaTimeSource for Stream Clock Sources

Property	Description
OcaString Role	Shall reflect the localized_description field of the CLOCK_SOURCE descriptor.
OcaString Label	Shall reflect the object_name field of the CLOCK_SOURCE descriptor.
OcaTimeSourceAvailability Availability	No Milan-specific requirements.
OcaTimeDeliveryMechanism TimeDeliveryMechanism	Shall always be StreamEndpoint.
OcaSDPString ReferenceSDPDescription	No Milan-specific requirements.

Property	Description
OcaString ReferenceID	No Milan-specific requirements.
OcaTimeSourceSyncStatus SyncStatus	No Milan-specific requirements.
OcaParameterRecord TimeDeliveryParameters	Shall be of type OcaTimeDeliveryParameters_StreamEndpoint as defined in [AES70-2A].

7.6.3. Unsupported OcaTimeSource methods

The following OcaTimeSource are not supported by this Adaptation. These methods shall be defined in the Device API, but when called shall return the OcaStatus value NotImplemented:

- SetTimeDeliveryMechanism(...)
- SetReferenceType(...)
- SetTimeDeliveryParameters(...)

7.7. Clocking scenarios (informative)

This Adaptation supports several different clocking scenarios, depending on the configuration of Asynchronous Sample Rate Converters (ASRCs). These are shown in Table 36.

Table 36 shows only basic cases with one set of identical Stream Inputs, one Audio Unit, and one set of identical Stream Outputs. More complex scenarios may be constructed by combining elements of these cases.

In reading this table, it may be helpful to refer to Figure 3.

Table 36. Basic clocking scenarios

Case	ASRC present for		OcaMediaClock3 objects present		Description
	Input	Output	Main	Output	
1	N	N	Y	N	Main clock is used for input, processing, and output.
2	Y	N	Y	N	Input stream need not be synced to device. Main clock is used for processing and output.
3	N	Y	Y	Y	Main clock is used for input and processing. Separate output clock is used for output.
4	Y	Y	Y	N	Input stream need not be synced to device. Main clock is used for processing. Separate output clock is used for output.

This Adaptation does not specify the effect of receiving an input stream with incompatible clocking. Incompatible clocking means:

- (Cases 1 and 3) Input stream samples are not synced to the device main clock.
- (Cases 2 and 4) Input stream sampling parameters exceed the conversion capabilities of the input ASRC.

Annex A. (Informative) Milan Entity model mappings

This Annex describes the mapping of Milan Entity Model elements to CM4 elements. It is an inverted version of the normative text in Clause 7, which specifies the mapping of CM4 elements to Milan Entity Model elements.

A.1. Entity

A Milan Entity is the top level of the Entity Model. It contains the Configurations (Clause A.2) that are defined for the Milan Device.

In this Adaptation, a Milan Entity does not correspond to any one AES70 object or group of objects; rather, it contains all the Milan Descriptors of the entire Device.

The value of the ENTITY Descriptor field entity_id is always nonempty, in order to enable construction of the OcaMediaTransportEndpoint.IDExternal field for all Endpoints. This field is the unique identifier of an OcaMediaTransportEndpoint instance across the network. See Clause 7.3.6.2.

A.2. Configuration

A Milan Configuration is defined by [ATDECC(ATDECC Entity Model - Overview)] as follows:

> "A Configuration describes one operating mode of the ATDECC Entity. It contains all of the Streams, Units, Clock Sources, Interfaces, Jacks and entity level Controls."

In Milan, a number of Configurations can exist in a Device, each representing a static configuration of the Entity. Selecting Configurations or switching between them is outside the scope of this Adaptation.

Elements of Configurations are described in the following sections.

A.3. Mapping of Milan Configuration elements to CM4

The mapping of Milan Configuration elements to CM4 is illustrated in Figure 5 and described in the subsections following. Figure 5 illustrates the reverse of the mapping illustrated in Figure 3 in Clause 7.1, which shows the mapping of CM4 elements to Milan Configuration elements.

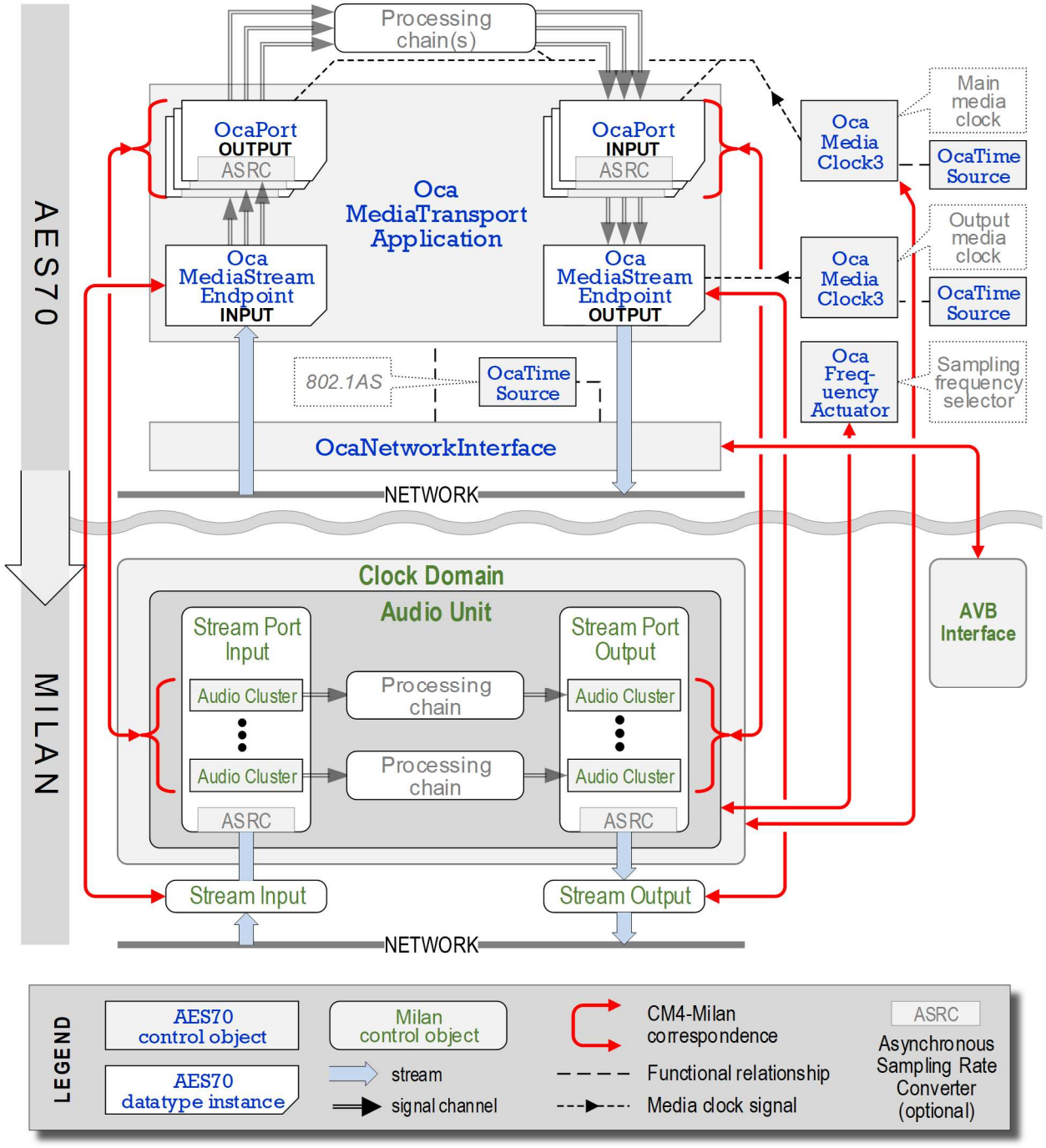

Figure 5. Milan->CM4 mapping
Showing the use case with one Clock Domain and one Audio Unit.

A.3.1. Stream Inputs and Stream Outputs

A Stream Input models the ingress of a stream from an AVB network to an Entity. A Stream Output models the egress of a stream from an Entity to an AVB network.

In the OcaMediaTransportApplication object, each Stream Input and each Stream Output is represented by an OcaMediaStreamEndpoint element collected by the Endpoints property (Clause 7.3.6), an entry in the EndpointStatuses list (Clause 7.3.7), and a Counterset inside the EndpointCounterSets property (Clause 7.3.8).

A Stream Input is represented by an input OcaMediaStreamEndpoint instance; a Stream Output is represented by an output OcaMediaStreamEndpoint instance.

A.3.2. Audio Units

An Audio Unit is a Milan control abstraction that represents one or more audio processing chains that share a common Clock Domain. Multiple Audio Units can share a single Clock Domain. In a Device that supports multiple Clock Domains, there will be at least one Audio Unit for each Clock Domain.

In this Adaptation, every Audio Unit is associated with an OcaMediaTransportApplication object. Usually, there will be one such object for all Audio Units in the Device. In that object, each Audio Unit is described by an entry in the list OcaMediaTransportApplication.AdaptationData.AudioUnits.

Each entry in the AudioUnits list is an Audio Unit descriptor of datatype MilanAudioUnit - see Clause 7.3.2.1.

When an OcaMediaTransportApplication object supports multiple Audio Units, each Audio Unit has its own discrete subset of the object's OCA Ports. See Clause 7.3.4 for details.

Each Audio Unit is associated with an OcaMediaClock3 object that identifies the clock synchronization source and specifies the set of available sampling rates. The actual sampling rate used is specified by an associated OcaFrequencyActuator object. See Clause 7.5.1 and Figure 5.

A.3.3. Stream Ports

A Stream Port models an ingress or egress point of a stream to an Audio Unit.

- A Stream Port Input models an ingress point through which a stream is routed from an AVB network to one or more processing chains in the containing Audio Unit.
- A Stream Port Output models an egress point through which a stream is routed from one or more processing chains in the containing Audio Unit to an AVB network.

Stream Ports themselves have no direct CM4 counterparts. However, Stream Ports contain Audio Clusters (see Clause A.3.4) which are analogous to OCA Ports of the associated OcaMediaTransportApplication object. This is illustrated in Figure 5, which shows a Stream Input Port and a Stream Output Port with no explicit relationships to CM4, but whose Audio Units correspond to OCA Ports in the OcaMediaTransportApplication object.

Stream Ports also may contain Audio Maps, which are analogous to CM4 channel maps stored in ChannelMap properties of OcaMediaStreamEndpoint instances. See Clause A.3.5 for a complete description of Milan channel mapping and its linking to CM4 elements.

A.3.4. Audio Clusters

An Audio Cluster is a Milan control abstraction belonging to a Stream Port. Each Audio Cluster describes a single signal channel in the Stream Port's Stream. Every channel in an AAF Stream, except for channels that are ignored by the Device, belongs to an Audio Cluster.

An audio processing chain connects to one or more Audio Clusters at its input and to one or more Audio Clusters at its output.

In this Adaptation, each Audio Cluster is represented by an OCA Port in the associated OcaMediaTransportApplication object. See Clauses 7.3 and A.3.3.

In the OcaMediaTransportApplication.PortClockMap property, all of the entries for the OCA Ports of a given Stream Port share the same value of the SRCType field.

A.3.4.1. Port direction differences

The signal-flow direction of a Stream Port is defined relative to the audio processing chains in the enclosing Audio Unit. Thus, a Stream Port Input contains Audio Clusters that define the inputs to the Audio Unit's processing chains, while a Stream Port Output contains Audio Clusters that define the outputs of those processing chains.

In contrast, AES70 audio processing is ***external*** to the OcaMediaTransportApplication object. This difference causes the directional sense of the OCA Ports to be reversed, as follows (see Figure 5):

- An Audio Cluster that defines a processing chain ***input*** is mapped to an AES70 ***Output*** Port that models an output from the OcaMediaTransportApplication object to the audio processing chain.
- An Audio Cluster that defines a processing chain ***output*** is mapped to an AES70 ***Input*** Port that models an input from the processing chain to the OcaMediaTransportApplication object.

A.3.5. Channel mapping

A Stream Port (i.e. a Stream Port Input or Stream Port Output) controls the mapping (routing) of Stream channels to and from signal channels in Audio Clusters.

Stream Port channel mappings may be statically or dynamically managed. Static channel mappings are set at time of manufacture, and are described by Audio Maps - see Clause A.3.5.1. Dynamic mappings are managed by AVDECC control API commands.

- A Stream Port Output's channel mappings can be statically or dynamically managed.
- A Stream Port Input's channel mappings are always dynamically managed.

When a Stream Port Output uses dynamic mapping, there is no associated Audio Map and the field ChannelMapDynamic of the associated OcaMediaStreamEndpoint instance is set to TRUE. Otherwise, there is an associated Audio Map, and ChannelMapDynamic is set to FALSE.

For Stream Port Inputs , there is always an associated Audio Map, and ChannelMapDynamic is always set to FALSE.

CM4's channel map is in the property OcaMediaStreamEndpoint.ChannelMap. This Adaptation requires compliant Devices to maintain ChannelMap at all times, regardless of whether the mapping is statically stored in an Audio Map or dynamically maintained via a Milan control API. This is specified normatively in Clause 7.3.6.4.

A.3.5.1. Audio Maps

An Audio Map is a Stream Port element that specifies the mapping of a stream's signal channels to the internal signal channels of one or more Audio Clusters. Each Audio Map is a static element defined by an AUDIO_MAP descriptor linked to a STREAM_PORT_INPUT or STREAM_PORT_OUTPUT instance.

A.3.5.2. Channel mapping location differences

In Milan, channel mapping functions reside in Stream Ports. In contrast, CM4 channel mapping functions reside in OcaMediaStreamEndpoint instances. The difference is illustrated in Figure 5, which shows the different elements that convert streams to channels, and vice versa.

A.4. Other Configuration elements

In addition to the stream processing elements described above, a Configuration can contain other kinds of elements. The full set of these is specified in [ATDECC]; those relevant to this Adaptation are noted below.

A.4.1. Clock Domain

A Clock Domain is the source of a common clock signal within an Entity. Each Clock Domain is represented by an OcaMediaClock3 object as specified by Clause 7.5.1.

A.4.2. Clock Source

A Clock Source is a time reference that a Clock Domain can use. Each Clock Source is represented by an OcaTimeSource object as specified by Clause 7.6.2.

- The case of internal Clock Sources is covered in Clause 7.6.2.1.
- The case of external Clock Sources is covered in Clause 7.6.2.2
- The case in which a Clock Source is derived from an input Stream is covered in Clause 7.6.2.3.

A.4.3. AVB Interface

An AVB Interface is the Milan control model for a physical network interface with AVB functionality. In this Adaptation, it is represented by an OcaNetworkInterface object, as specified by Clause 7.2.

An AVB Interface contains an IEEE-802.1AS time receiver. In this Adaptation, the time receiver's IEEE-802.1AS-related parameters are modelled by an OcaTimeSource object, as specified by Clause 7.6.1.

Annex B. (informative) Identify control

The Milan Identify control is a Device user interface (UI) element that assists users in the physical identification of Devices on an AVB network.

An Identify UI control can be one of three types:

A. a UI indicator on the Device that Controllers can activate to signal the Device's physical presence; or conversely
B. a UI control on the Device that users can activate to signal the Device's network presence to a Controller; or
C. a combination indicator/control that does both of the above.

Identify controls are not connection management elements and are not required by this standard. This informative Annex is included for completeness, because the Milan specification describes them.

An Identify control can be appropriately represented by an AES70 control object whose class is as follows:

- Identity control is a UI indicator (type A above) ..... OcaIdentificationActuator
- Identity control is a UI control (type B above) ..... OcaIdentificationSensor
- Identity control is a combination (type C above) .....OcaIdentificationActuator

In the case of type C, the OcaIdentificationActuator's Active property will be linked to the state of the UI control; the Controller can subscribe to this property and will receive a Notification when that control's state changes, i.e. when the user activates it. Thus, the actuator can also behave as a sensor, thereby supporting the bidirectional function of the control.

Annex C. (Informative) Example Device configurations

C.1. Example: Microphone

This example is analogous to the simple microphone example given in [AVnu-Milan]. The Device contains a microphone and an Ethernet interface. Its main usage is to stream captured audio on an AVB network through a 48kHz 1-channel AAF Stream.

The CM4 model is illustrated in Figure 6, which shows the following elements:

- One instance of **OcaMediaTransportApplication**, containing one input **OcaMediaStreamEndpoint** supporting the CRF format, and one output **OcaMediaStreamEndpoint** supporting the AAF format at 48kHz with 1 channel.
- One instance of **MilanOcaMediaTransportSessionAgent**, containing a single **OcaMediaTransportSession** and one **OcaMediaTransportSessionConnection**.
- One instance of each **OcaNetworkInterface** and **OcaTimeSource**, both representing the AVB Interface of the Device.
- One instance of **OcaMediaClock3** with its sample rate selection limited to 48kHz, as per the Audio Unit being modelled.
- Two instances of **OcaTimeSource**, each representing an available Clock Sources The Device is able to either use its internal media clock or derive it from the input CRF Stream.
- One Input **OcaPort**, attached to the **OcaMediaTransportApplication** object. The Output Endpoint's **ChannelMap** maps the single AAF channel to this **OcaPort** statically; the Endpoint's **ChannelMapDynamic** parameter is **FALSE**.

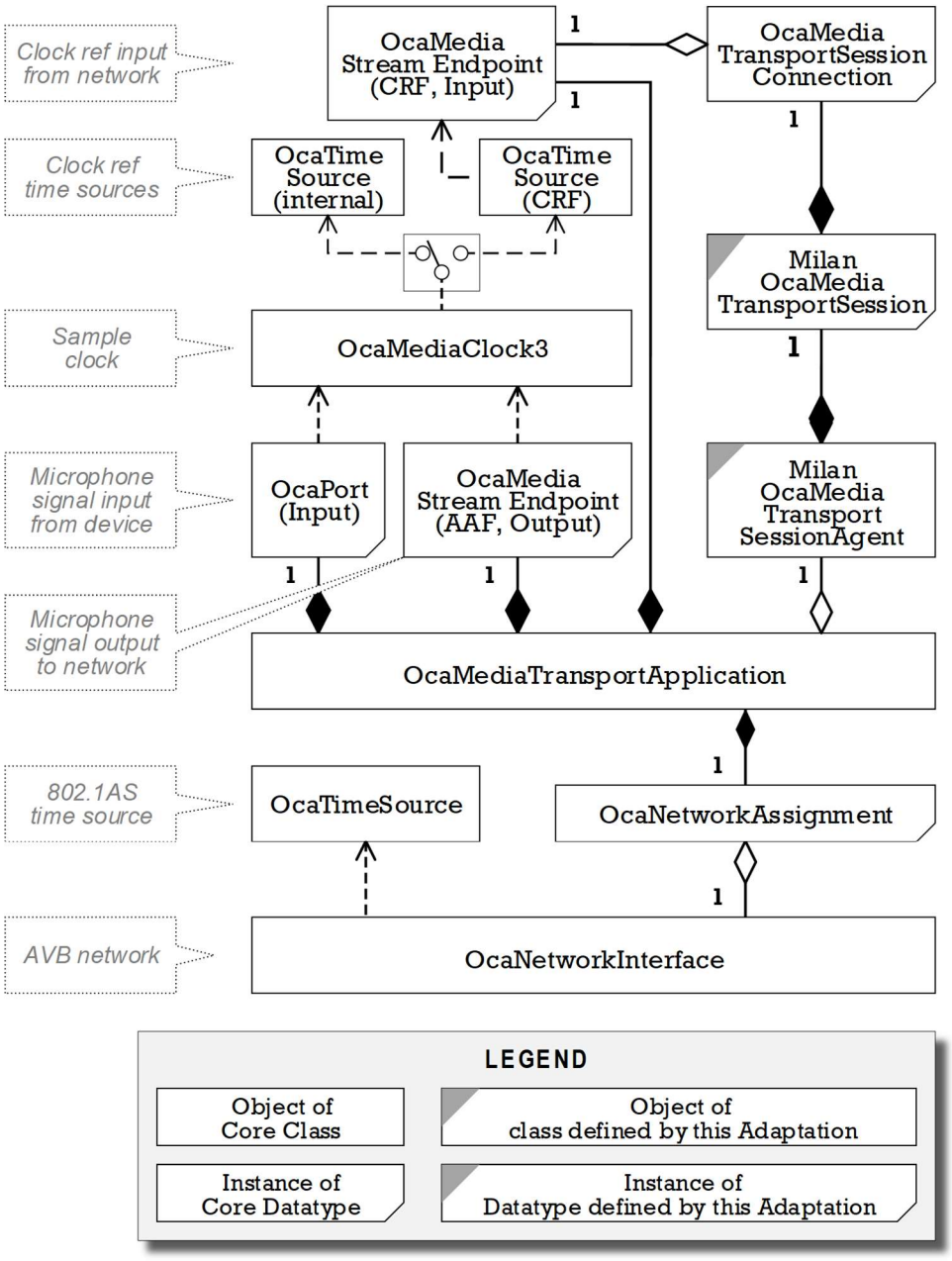

Figure 6. Example CM4 model: Microphone

C.2. Example: Simple speaker

This example is analogous to the simple speaker example given in [AVnu-Milan]. The Device contains a loudspeaker and an Ethernet interface. Its main usage is to render one audio channel from an input 48kHz AAF Stream of 1 to 8 channels.

The CM4 model is illustrated in Figure 7, which shows the following elements:

- One instance of **OcaMediaTransportApplication**, containing one input **OcaMediaStreamEndpoint** supporting the AAF format at 48kHz with 1 to 8 channels.
- One instance of **MilanOcaMediaTransportSessionAgent**, containing a single **OcaMediaTransportSession** and one **OcaMediaTransportSessionConnection**.
- One instance of each **OcaNetworkInterface** and **OcaTimeSource**, both representing the AVB Interface of the Device.
- One instance of **OcaMediaClock3** with its sample rate selection limited to 48kHz, as per the Audio Unit being modelled.
- One instance of **OcaTimeSource**, which represents the derivation of the Clock Source from the input AAF Stream.
- One Output **OcaPort**, attached to the **OcaMediaTransportApplication** object. The Endpoint's **ChannelMap** maps one of the 1 to 8 AAF channel to this **OcaPort** dynamically; the Endpoint's **ChannelMapDynamic** parameter is **TRUE**.

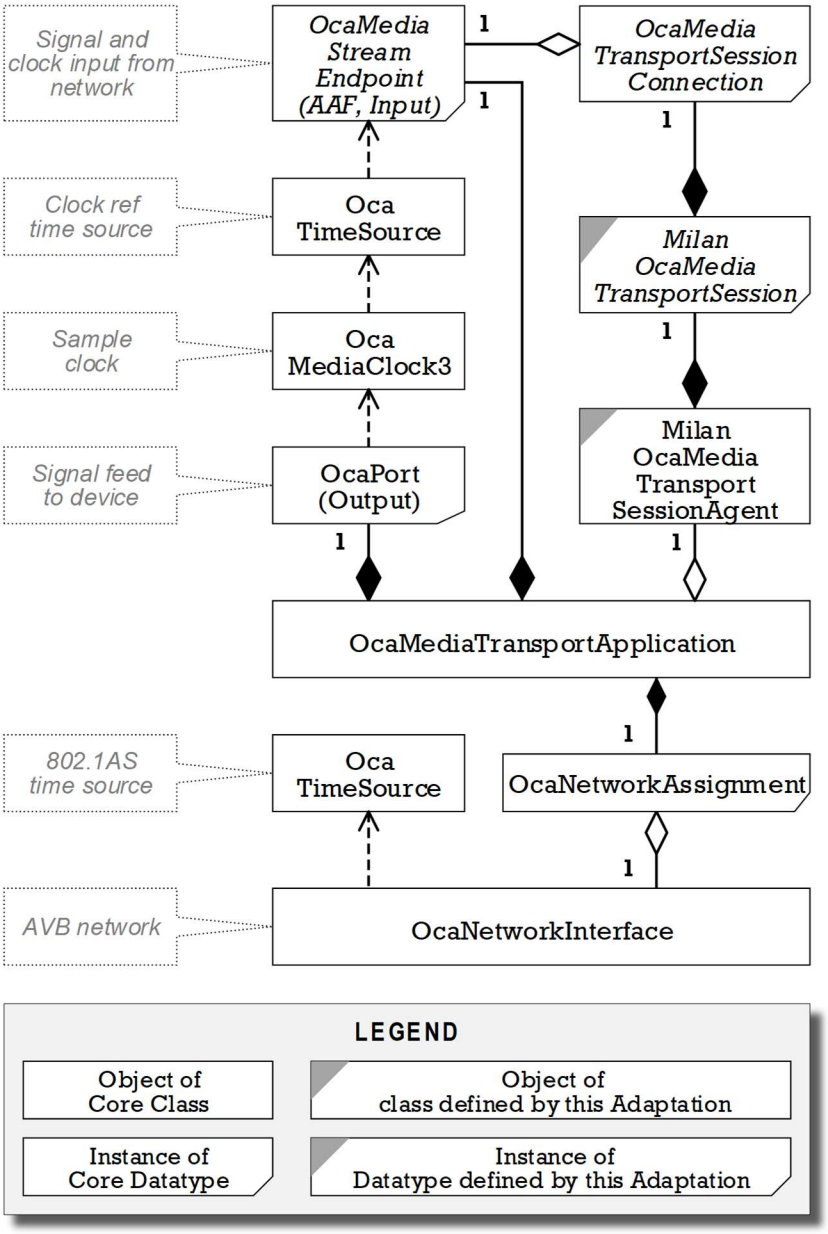

Figure 7. Example CM4 model: Simple speaker

C.3. Example: AES3 Break-Out box, redundant

This example is analogous to the redundant Break-Out box example given in [AVnu-Milan]. The Device is composed of an XLR connector for AES3 output and two Ethernet interfaces. Its main usage is to convert the input 48kHz AAF Stream (1 to 8 channels) to a 48kHz 2-channel AES3 audio output. Redundancy is provided by the two Ethernet ports; one is plugged to the primary network and the other to the secondary network.

The CM4 model is illustrated in Figure 8, which shows the following elements:

- One instance of **OcaMediaTransportApplication**, containing a pair of redundant Input Endpoints supporting the AAF format at 48kHz with 1 to 8 channels.
- One instance of **MilanOcaMediaTransportSessionAgent**, containing two Sessions with one Connection each.
- Two instances each of **OcaNetworkInterface** and **OcaTimeSource**, representing the primary and the secondary AVB Interfaces of the Device.
- Two instances of **OcaTimeSource**, representing the derivation of the Clock Sources from the input AAF Streams.
- One instance of **OcaMediaClock3** with its sample rate selection limited to 48kHz, as per the Audio Unit being modelled. This instance can use either input stream **OcaTimeSource** instance.
- Two Output **OcaPorts**, both attached to the **OcaMediaTransportApplication** object. Each **OcaPort** is dynamically mapped to the Stream channels of both primary and secondary Input Endpoints; those Endpoints' **ChannelMapDynamic** parameters are **TRUE**.

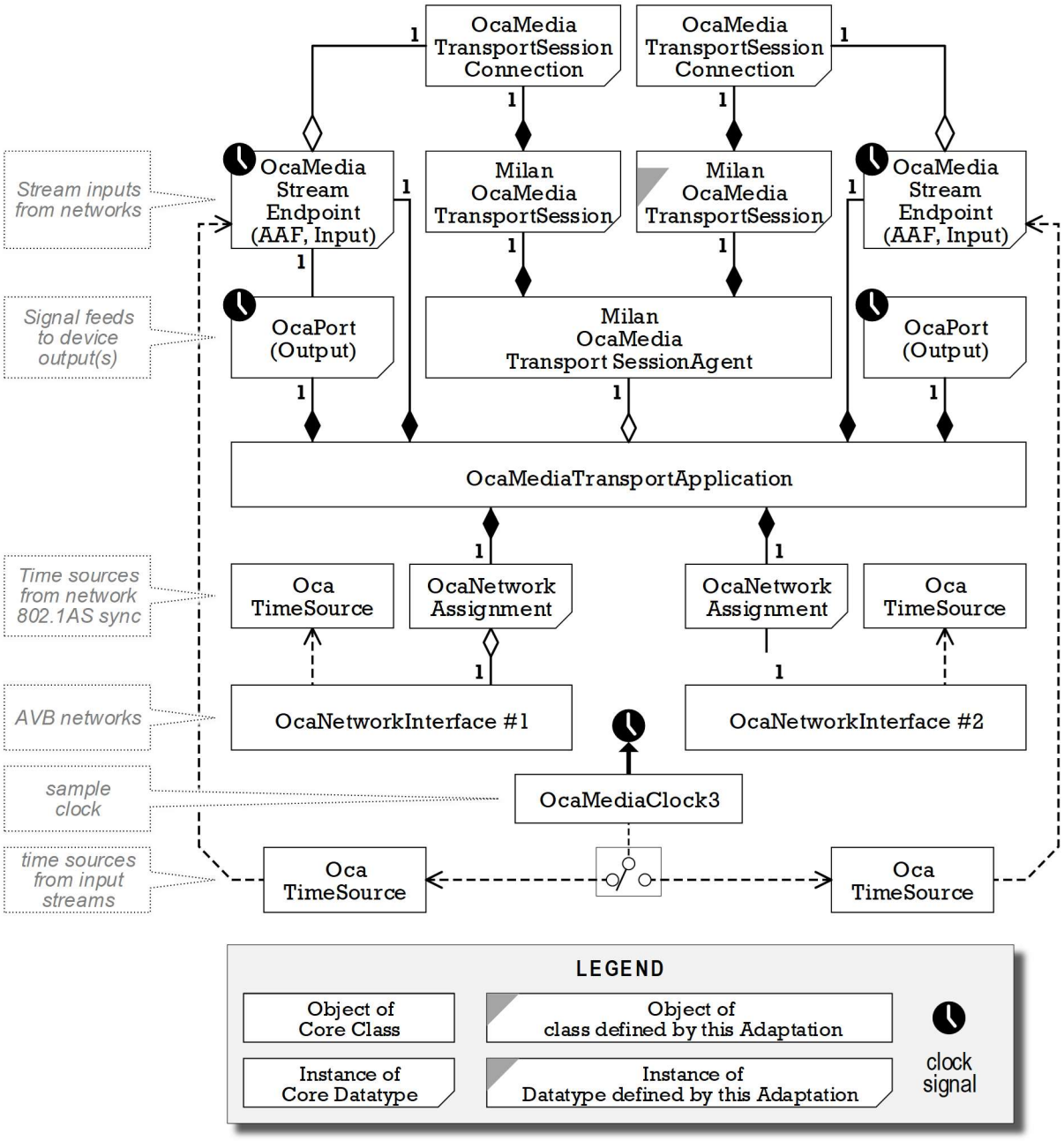

Figure 8. Example CM4 model: AES breakout box, redundant

C.4. Example: Four-channel line amplifier

This example is analogous to the simple 4-channel amplifier example given in [AVnu-Milan]. The Device is composed of four XLR connectors for line-level analog outputs and an Ethernet interface. Its main usage is to get four audio channels from the input AAF Stream (48kHz or 96kHz, 1 to 8 channels), process and amplify them.

The CM4 model is illustrated in Figure 9 which shows the following elements:

- One instance of **OcaMediaTransportApplication**, containing one input **OcaMediaStreamEndpoint** supporting the AAF format at 48kHz or 96kHz, with 1 to 8 channels.
- One instance of **MilanOcaMediaTransportSessionAgent**, containing a single **OcaMediaTransportSession** and one **OcaMediaTransportSessionConnection**.
- One instance each of **OcaNetworkInterface** and **OcaTimeSource**, both representing the AVB Interface of the Device.
- One instance of **OcaTimeSource** that represents the derivation of the Clock Source from the input AAF Stream.
- One instance of **OcaMediaClock3** with its sample rate selection limited to 96kHz, as per the Audio Unit being modelled. This instance references the one instance of **OcaTimeSource**, which represents the single available Clock Source.
- Four Output **OcaPorts**, all attached to the **OcaMediaTransportApplication** object. The Endpoint's **ChannelMap** maps the 1 to 8 AAF channels to these **OcaPorts** dynamically, as characterized by the Endpoint's **ChannelMapDynamic** parameter; the Endpoint's **ChannelMapDynamic** parameter is **TRUE**

Furthermore, there is a Sample Rate Converter in the Stream Port Input capable of converting 48kHz input streams to 96kHz, to allow internal processing at 96kHz This converter is shown as part of the **OcaMediaStreamInput**. The **PortClockMap** of the **MilanOcaMedia-TransportApplication** specifies **Synchronous** SRC for all four of its entries.

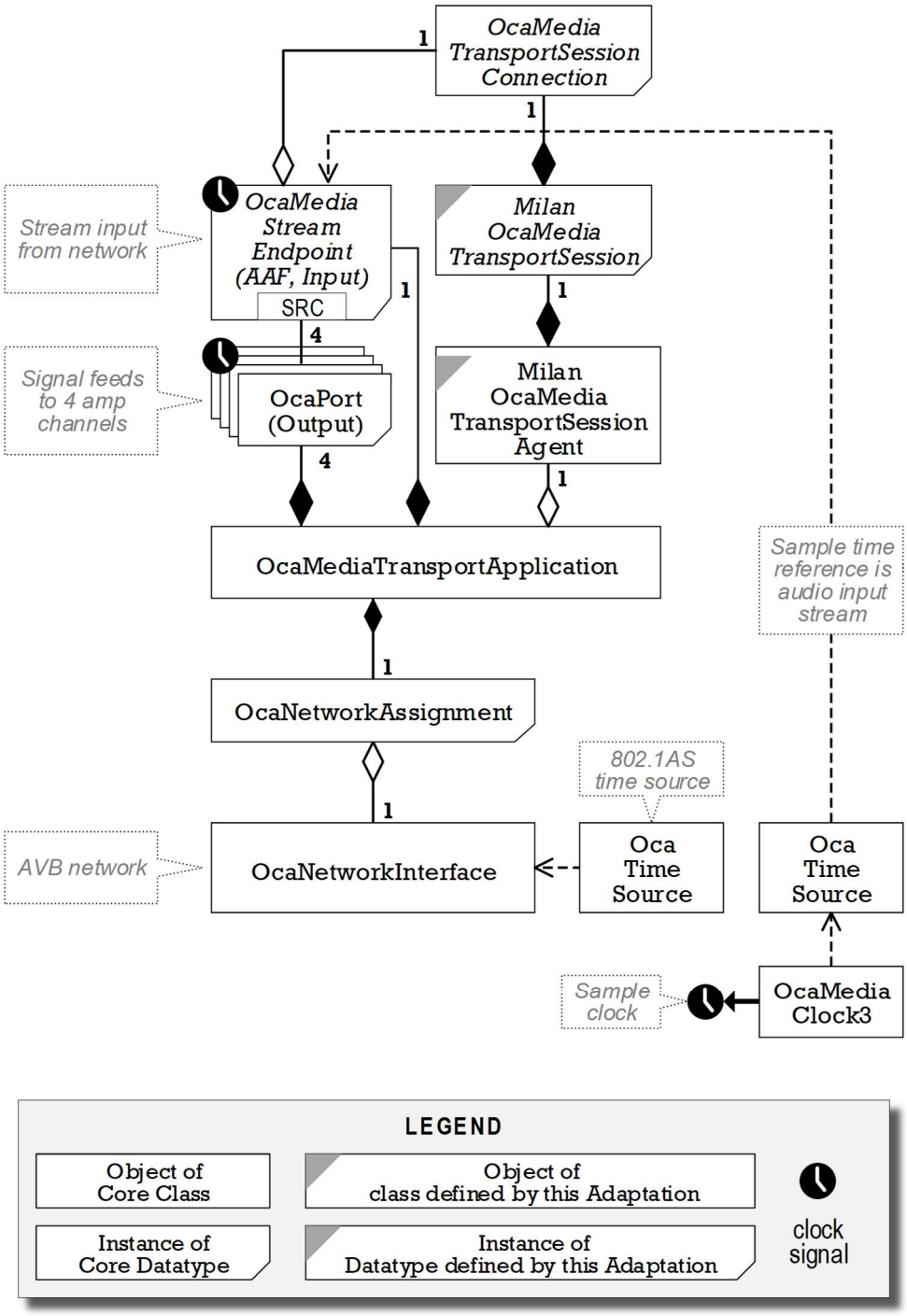

Figure 9. Example CM4 model: Four-channel line amplifier

C.5. Example: DSP matrix

This example is analogous to the pure 4x4 DSP unit example given in [AVnu-Milan]. The unit is a DSP matrix that resides on an AVB network, accessed via an Ethernet interface. Its purpose is to acquire four audio channels from two input AAF Streams (48kHz, 1 to 8 channels), mix them, and output them to a single output AAF Stream (1 to 4 channels).

The CM4 model is illustrated in Figure 10, which shows the following elements:

- One instance of **OcaMediaTransportApplication**, containing three Input and two Output Endpoints, as follows:
  - Two Input Endpoints that carry audio and support the AAF format at 48kHz with 1 to 8 channels.
  - One additional Input Endpoint that supports the CRF format.
  - One Output Endpoint that carries audio and supports the AAF format at 48kHz with 1 to 4 channels.
  - One additional Output Endpoint that supports the CRF format.
- One instance of **MilanOcaMediaTransportSessionAgent**, containing three Sessions with one Connection each.
- One instance each of **OcaNetworkInterface** and **OcaTimeSource**, both representing the AVB Interface of the Device.
- Two instances of **OcaTimeSource**, one representing the Clock Source of the CRF input stream, the other representing an internal Clock Source.
- One instance of **OcaMediaClock3** with its sample rate selection limited to 48kHz, as per the Audio Unit being modelled. This instance can use either the CRF or the internal **OcaTimeSource** instance.
- Four Input **OcaPorts** and four Output **OcaPorts**, all attached to the **OcaMediaTransportApplication** object. The **ChannelMaps** of the two Input AAF Endpoints dynamically map their Stream channels to the four output **OcaPorts**; the **ChannelMap** of the Output AAF Endpoint statically maps its Stream channels to the four input **OcaPorts**.

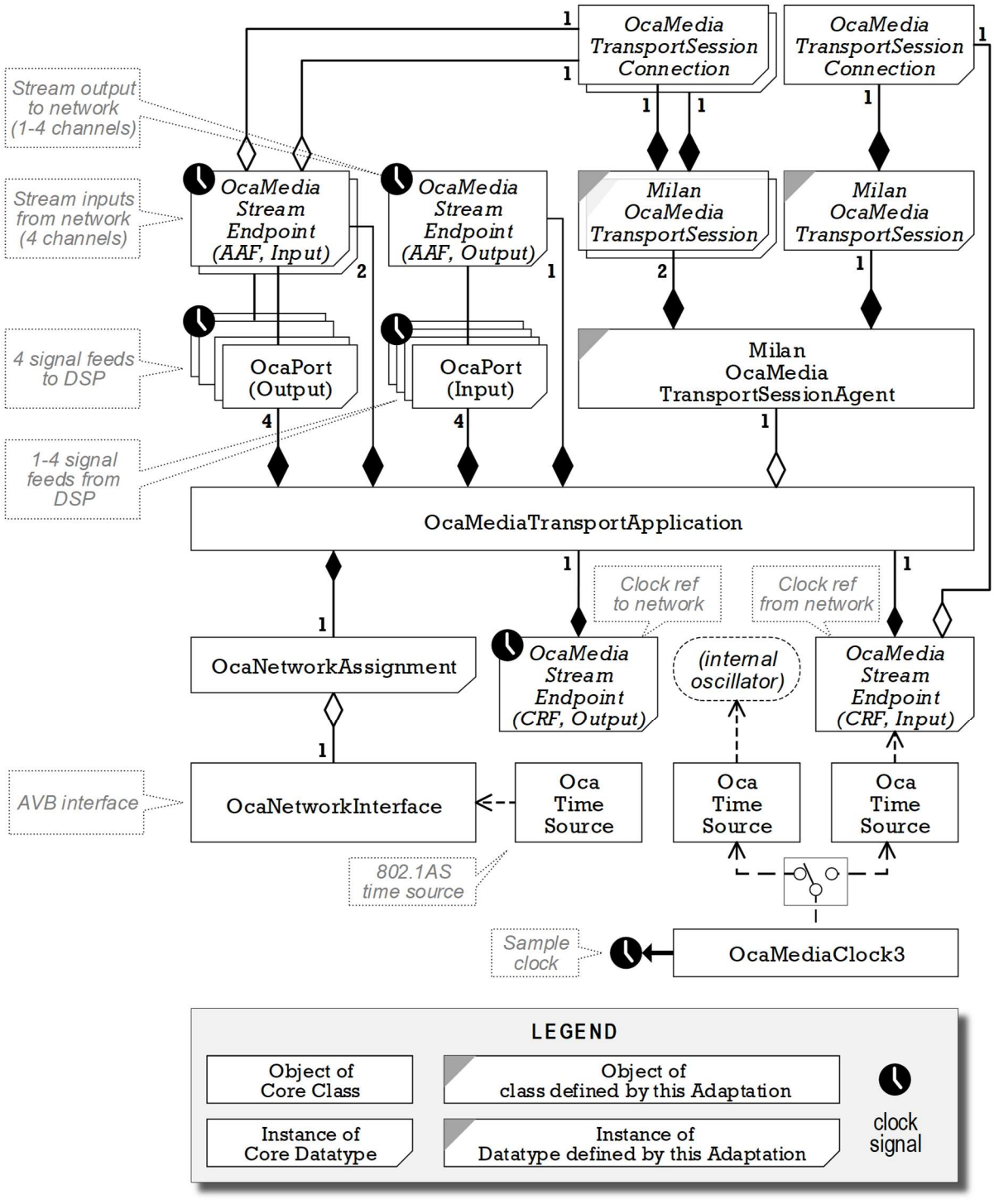

Figure 10. Example CM4 model: DSP matrix

Annex D. (Informative) Stream Formats

Table 37 lists all stream formats supported in this Adaptation – which is limited to the set of format strings originally specified in [AVnu-Milan] – and the field values of the equivalent OcaMediaStreamMode as specified by [AES70-2A].

Table 37. AAF Stream Format String values and corresponding Stream Mode parameters

Stream format string	Frame Format	Encoding Type	Sampling Rate	Channel Count	Packet Time
0x0205022000406000	AAF	"audio/L32"	48 kHz	1	125us
0x0205022000806000	AAF	"audio/L32"	48 kHz	2	125us
0x0205022001006000	AAF	"audio/L32"	48 kHz	4	125us
0x0205022001806000	AAF	"audio/L32"	48 kHz	6	125us
0x0205022002006000	AAF	"audio/L32"	48 kHz	8	125us
0x020702200040C000	AAF	"audio/L32"	96 kHz	1	125us
0x020702200080C000	AAF	"audio/L32"	96 kHz	2	125us
0x020702200100C000	AAF	"audio/L32"	96 kHz	4	125us
0x020702200180C000	AAF	"audio/L32"	96 kHz	6	125us
0x020702200200C000	AAF	"audio/L32"	96 kHz	8	125us
0x0209022000418000	AAF	"audio/L32"	192 kHz	1	125us
0x0209022000818000	AAF	"audio/L32"	192 kHz	2	125us
0x0209022001018000	AAF	"audio/L32"	192 kHz	4	125us
0x0209022001818000	AAF	"audio/L32"	192 kHz	6	125us
0x0209022002018000	AAF	"audio/L32"	192 kHz	8	125us