GY

中华人民共和国广播电视行业标准

GY/T 322.1—2019

网络音频应用的开放式控制架构 第1部分：框架

Audio applications of networks - open control architecture—
Part 1: Framework

2019－04－28 发布　　2019－04－28 实施

国家广播电视总局　　发布

目　　次

前　言

GY/T 322《网络音频应用的开放式控制架构》分为以下三部分：

——第1部分：框架；

——第2部分：类结构；

——第3部分：用于TCP/IP网络的协议。

本部分为GY/T 322的第1部分。

本部分按照GB/T 1.1—2009给出的规则起草。

本部分是参照AES70-1-2015《网络音频应用的开放式控制架构　第1部分：框架》编制的。

请注意本文件的某些内容可能涉及专利。本文件的发布机构不承担识别这些专利的责任。

本部分由全国广播电影电视标准化技术委员会（SAC/TC 239）归口。

本部分起草单位：中央广播电视总台、国家广播电视总局广播电视科学研究院、国家广播电视总局广播电视规划院、江苏省广播电视总台、浙江广播电视集团、苏州市福川科技有限公司、北京英夫美迪科技股份有限公司、北京众和传新科技有限公司、杭州联汇科技股份有限公司、上海佰贝科技发展有限公司、北京捷成世纪科技股份有限公司、苏州大学。

本部分主要起草人：钱岳林、朱峰、罗攀、潘宇、张磊、王兰岚、庞超、唐峰、张伟、邓向冬、董升来、何晶、孙岩君、李维民、陈武、董晓坡、陈沁、唐卫平、陈立德、赵崇峰、肖仲喆。

引　言

0.1　概述

开放式控制架构（OCA）规定了用于专业媒体网络监控的可扩展的控制协议架构。开放式控制架构仅涉及设备监控，没有定义用于传送流媒体或描述媒体内容的标准。只要底层的通信网络能够承载开放式控制架构的监控流量，它就可以与任何流媒体传输协议一起使用集成。

开放式控制架构未提供完整的设备实现模型。开放式控制架构建立了设备的监控功能模型，并不包括其全部的信号路径。如果设备特定部分要素不具备远程可控的特征，则不必在该设备的符合开放式控制架构协议接口中表述。

开放式控制架构的第1部分是参照AES70-1-2015《网络音频应用的开放式控制架构　第1部分：框架》编制的，英文原文可从http://www.aes.org/publications/standards/search.cfm?docID=101下载。

开放式控制架构的第2部分定义了用于媒体网络监控的开放式控制架构的类结构。第2部分是参照AES70-2-2015《网络音频应用的开放式控制架构　第2部分：类结构》编制的，英文原文可从http://www.aes.org/publications/standards/search.cfm?docID=102下载。

开放式控制架构的第3部分是参照AES70-3-2015《网络音频应用的开放式控制架构　第3部分：用于TCP/IP网络的协议》编制的，英文原文可从
http://www.aes.org/publications/standards/search.cfm?docID=103下载。

0.2　架构目标与约束

开放式控制架构基于以下功能需求和要求：

a)　功能性

开放式控制架构支持以下功能：

1)　发现与网络连接的符合开放式控制架构的设备；
2)　对设备间的媒体流路径进行定义或取消定义；
3)　对符合开放式控制架构的设备进行控制操作和参数配置；
4)　对符合开放式控制架构的设备进行监测操作和参数配置；
5)　对于具有可重配置信号处理和/或控制能力的设备，定义和管理配置参数；
6)　升级受控设备的软件和固件。包括故障安全升级功能。

b)　安全性

开放式控制架构支持以下对数据控制和监测采取的安全措施：

1)　实体认证；
2)　防窃听；
3)　完整性保护；
4)　新鲜度—这里的“新鲜度”是指对重放攻击中的检测出的重放消息的真实性。

c) 可扩展性

开放式控制架构支持具有至少有 10000 个应用设备的网络。开放式控制架构对设备的物理分布施行最小限制。

d) 可用性

开放式控制架构通过提供以下特性来支持高可用性：

1) 对符合开放式控制架构的设备的设备管理；
2) 对与符合开放式控制架构的设备相连的网络进行管理；
3) 在错误和配置更改后，进行高效的网络重新初始化。

e) 鲁棒性

开放式控制架构通过提供以下特性来支持鲁棒性：

1) 操作确认机制；
2) 处理控制数据丢失的机制；
3) 处理符合开放式控制架构的设备失效的机制；
4) 网络实施者可使用的网络鲁棒性机制的建议。

f) 安全标准

开放式控制架构允许实现符合生命安全紧急标准的媒体网络。

g) 兼容性

在开放式控制架构的发展过程中将最大限度地实现不同版本之间的兼容性。基于开放式控制架构的一个版本的控制器将基于下列方式在另一个版本的符合开放式控制架构的设备上运行：

1) 对于基于旧版本的设备，新版本的控制器按照与该设备相同版本进行工作；
2) 对于基于更新版本的设备，旧版本的控制器能够监控设备中与控制器相同版本的所有功能，且不妨碍仅在设备版本中定义的功能。

h) 可分析性

开放式控制架构定义了允许访问以下信息的诊断功能：

1) 每个设备的所有组件、硬件和软件的版本信息；
2) 设备的网络参数（如 MAC 地址，IP 地址）；
3) 设备状态（包括设备网络接口状态）；
4) 媒体流参数（用于设备的每个活动的接收和/或发送媒体流）；
5) 通信错误。

网络音频应用的开放式控制架构
第1部分：框架

1 范围

GY/T 322的本部分规定了网络音频应用的开放式控制架构的模型和机制。这些模型和机制共同组成了开放式控制架构的框架。

本部分适用于对网络音频应用的监控，不适用于传送流媒体或描述媒体内容。

2 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅注日期的版本适用于本文件。凡是不注日期的引用文件，其最新版本(包括所有的修改单)适用于本文件。

GB/T 13000—2010 信息技术 通用多八位编码字符集（UCS）（ISO/IEC 10646:2003，IDT）

GY/T 322.2—2019 网络音频应用的开放式控制架构 第2部分：类结构

GY/T 322.3—2019 网络音频应用的开放式控制架构 第3部分：用于TCP/IP网络的协议

3 术语、定义和缩略语

3.1 术语和定义

下列术语和定义适用于本文件。

3.1.1

控制器 controller

通过一个符合开放式控制架构的接口来监控设备的联网的软件单元。

注：控制器可托管在专用计算机中，也可是运行于某个设备或某些其他环境中的软件单元。

3.1.2

控制协议 control protocol

用于远程监控网络设备应用功能的应用协议。

3.2 缩略语

下列缩略语适用于本文件。

OCA 开放式控制架构（Open Control Architecture）

ONo 对象编号（Object Number）
PDU 协议数据单元（Protocol Data Unit）
VCA 压控放大器（Voltage-controlled Amplifier）

4 顶层设计

4.1 概述

开放式控制架构支持在应用层对符合开放式控制架构的设备进行监控。它不提供音视频信号传输，而需与各种音视频信号的传输方案一起集成使用。

开放式控制架构规定的协议具有可扩展性，可有序的合并新的设备类型和设备升级，支持各种媒体网络中的功能的以向上兼容的方式改进升级。

开放式控制架构适用于专业媒体网络。对于不大于100个节点的简单网络，在使用推荐的交换机时，设置过程不宜要求技术人员具备高级网络知识。具体要求如下：

a） 开放式控制架构的网络可使用符合工业标准的数据网络设备来运行；
b） 符合开放式控制架构的设备可与不符合开放式控制架构的设备无害共存；
c） 开放式控制架构的网络可以根据产品及应用的需要，在安全或不安全模式下运行。

4.2 面向对象

4.2.1 概念

开放式控制架构将通信设备的控制接口描述为对象的集合。每个对象是一个软件单元，它是特定类的实例化，并具有由该类定义的状态（称为属性）和程序化动作（称为方法）。

开放式控制架构的协议的所有动作和特征均以类的方式定义。协议的功能范围相当于它实现的类的功能库。类的集合完全决定了通信设备中可实例化什么类型的对象。

以这种方式定义的协议称为面向对象的协议，定义如下：

a） 类定义：定义可存在于设备中的对象类型；
b） 命名和寻址规则：定义如何识别对象及其属性；
c） 协议数据单元（PDU）的格式：指定发送和接收数据的实际格式；
d） 协议数据单元的交换规则：定义用于实现信息交换的通信序列（见 4.3）。

注：本部分以面向对象的设计术语表述，但并不要求使用面向对象的编程风格实现这些协议。实现时可选择基于对象或非基于对象的方式。

4.2.2 类

4.2.2.1 概述

4.2.2.1.1 类的内容

开放式控制架构的类均应包含以下内容：

a） 要素集合（属性、方法和事件，见 4.2.2.3）；
b） 唯一的类标识符；

c） 确切的父类（根类除外，因为它没有父类）。

注：标准的类名（见GY/T 322.2—2019）均以“0ca”开头。

4.2.2.1.2 继承

开放式控制架构的类应定义为层次树结构中的节点。该层次树结构应从单一的基础节点（根类）开始，并应按继承顺序排列。继承意味着类均从一个特殊性相对较弱的类（父类）派生出的特殊性更强的实体（子类）。一个类应展示（继承）其父类的所有特征，除非在该类定义中明确重写了原有的继承特征。

4.2.2.1.3 专有类

在开放式控制架构的生命周期中，往往会出现特定的产品，尤其是复杂的产品，需要用到专门的控制类，这些类不适合包含在标准的类树中，即专有类。可通过以下四个策略来实现：

a） 专有类可从任何的标准类或另外的专有类继承；

b） 专有类宜以最恰当明确的层级附加到标准类树中；

c） 对专有类应使用专有类的编号；

d） 专有类应根据 4.2.2.4 中所列的继承规则定义。

4.2.2.1.4 包含

在开放式控制架构中，一个类有时会包含另一个类，该包含类的对象合并了被它包含类的对象。

如果删除包含类的对象，也应删除被它包含的类的对象。

4.2.2.1.5 收集

在开放式控制架构中，一个类有时会收集另一个类，该收集类的对象引用了被它收集类的对象。

如果删除收集类的对象，不应删除被它收集的类的对象。

4.2.2.2 类标识符

4.2.2.2.1 概述

类标识符（类ID）应由谱系键值和版本号组成。

类ID表示为{n;i_1 • i_2 • i_3...}，其中n是类版本号，i_1 • i_2 • i_3...是谱系键值。

4.2.2.2.2 谱系键值

每个类均应用形如i_1•i_2•i_3...的层级键值来标识，其中i_1•i_2•i_3...应为类树的特定层级的兄弟节点中能唯一标识该类的正的非零整数值。i_1，i_2，i_3等被称为类索引。类索引值可非连续。

每个类的谱系键值均应由一组类索引组成。类索引从根类开始，向下延伸通过所有相关的子类，并结束于该类,这一组类索引标识了该类的完整谱系。这个键值可以根据描述层级的需要包含足够多的类索引。

示例：

对于谱系键值为1•2•12•7的类X，谱系键值应按如下方式从左到右解释：

1表示根类。

1•2表示根类的一个子类。

1•2•12表示父类是1•2的一个子类。

1•2•12•7表示类X，其父类为1•2•12。

4.2.2.2.3 标准类 ID

每个类的ID包括一个版本号，用于唯一标识类的修订版本。标准类的ID应依据GY/T 322.2—2019来分配，且通过不断地修订，GY/T 322.2—2019包含新的类以及现有类的新版本。

4.2.2.2.4 专有类 ID

专有类ID可由设备制造商设置。设备的制造商应在开放式控制架构设备管理单元（OcaDeviceManager）对象的属性中标识。包含专有对象的设备的控制器宜查询其对应的OcaDeviceManager对象，以获得设备制造商信息，并据此作出相应的操作。

注 1：两个制造商对不同的对象使用相同的专有类的索引值的现象是不可避免的。

如果专有类ID{n;i_1•i_2•i_3...i_k...}中某个特定的索引i_k是专有的，则其右侧的所有索引也应是专有的。

注 2：根据以上规则，标准类不应从专有类继承。

对专有类定义上的任何更改，应用更高的类版本号来标识。

4.2.2.3 类的属性、方法和事件

4.2.2.3.1 概述

开放式控制架构的类应具有允许访问其数据和操作状态的要素。类的要素有：

a) 属性：类应定义若干属性，代表类的可监控参数的变量，并能被控制网络所访问；
b) 方法：类应定义若干种方法，它们是一些程序，控制器可由符合开放式控制架构的协议命令来调用这些程序去获取和更改属性值、改变类的运行状态以及执行其他的操作；
c) 事件：类可以定义一个或多个事件，事件是由设备激发以通知控制器发生了特定的事件的回调函数。为了接收事件，控制器应事先订阅它们，见第6章。

这些要素应用于定义设备和控制器之间的数据交换协议。在通信协议的要素中表示类要素的方式取决于每个特定的符合开放式控制架构的协议，见GY/T 322.3—2019。

4.2.2.3.2 要素 ID

在类树中，每个属性、方法和事件不仅应分配一个名称，而且还要赋予一个要素ID，形如：LLtNN，其中：

LL应为两位数表示的类树层级。

例如，全局根类OcaRoot在层级01定义。OcaRoot的子类将在层级02定义。孙类将在层级03定义。等等。

t应为类型代码，用p表示属性，m表示方法，e表示事件。

NN应为每个类中的每个类型的序列号，起始值为01。

在每个类中，要素ID的值应是唯一的。

示例 1：

01p01 是定义在类树 01 层级上的类的第一个属性。在这种情况下，由于 OcaRoot 是层级 01 的唯一的类，因此 01p01 是 OcaRoot 的第一个属性。

示例 2：

03m02 是定义在类树的 03 层级上的类的第二个方法。在 03 层级上定义了多个类，此 ID 将适用于任何这些类的第二个方法。

注1：要素 ID 的规则旨在提供一种方法，用于唯一标识任何给定类的所有新增和继承要素的方法，并考虑到将来在不同层级上对类树进行扩充，同时不会带来标识的重复。

注2：关于这种方法的一个例子，参见附录 A 中的类 OcaGain。

注3：要素 ID 可以是属性 ID，方法 ID 或事件 ID，具体取决于其标识的要素的类型。

4.2.2.3.3 协议不变性

对于任何给定的类，无论使用哪种符合开放式控制架构的协议，以下项目应保持不变：

a） 属性、方法和事件集合；

b） 要素 ID 集合。

4.2.2.3.4 文本格式

开放式控制架构的属性、方法参数和事件参数中的所有文本均应采用UTF-8格式（见GB/T 13000—2010）。

4.2.2.4 类的继承和更新规则

继承规则确保通过从现有类创建新类，以不影响现有产品或系统操作的方式将新功能添加到协议中。继承能确保新的子类至少能支持其父类的功能和接口。

开放式控制架构的类继承规则是：

a） 除了根类之外的任何给定类，都应确切地从另一个类继承。

b） 子类应实现其父类所有的属性，方法和事件。

c） 子类可通过以下方式扩充父类的定义：

1） 添加新的属性、方法和/或事件；

2） 增强现有的属性、方法和/或事件的定义。在这种情况下，所增强的定义应支持由父类定义的所有功能。

d） 子类继承的方法和事件应保留其父类对应的要素 ID。

e） 标准类不应从专有类继承。

当更新现有类时，应遵守以下规则：

a） 类版本号应递增。

b） 更新的类应实现现有类的所有属性、方法和事件。

c） 更新的类可通过以下方式扩充现有类的定义：

1） 添加新的属性、方法和/或事件；

2） 增强现有属性、方法和/或事件的定义。在这种情况下，所增强的定义应支持由现有类定义的所有功能。

d） 被更新类的方法和事件应保留其父类对应的要素 ID。

4.2.3 类的实例化

正如创建其网络控制接口所需，设备应将类实例化为对象。每个对象应由唯一的对象编号（ONo）标识。

4.3 消息

4.3.1 概述

监控操作应由协议数据单元（PDU）中的消息来执行。PDU在两个对象之间传递，这两个对象可在不同设备中。开放式控制架构的消息应是以下三种类型之一：

a） 命令：请求设备返回数据和/或执行操作。

b） 应答：报告命令执行成功或失败。如果有，则返回所请求的数据。

c） 通知：报告设备内部发生的特定事件，并提供相关数据。

除了设备复位消息（见第12章）外，开放式控制架构的命令消息均应由对应的应答消息应答。通知消息不应响应。

4.3.2 消息分发服务

开放式控制架构支持两种消息分发服务：可靠服务和快速服务。

可靠分发服务应使用网络提供的有保证的传输手段。可靠服务不应用于组播。例如，在TCP/IP网络中，可靠服务使用TCP。

快速分发服务可使用较低开销，较低可靠性的传输手段。如果网络支持，快速服务可用于组播。例如，在TCP/IP网络中，快速服务使用UDP。

开放式控制架构的所有命令和应答消息应使用可靠分发服务来传递。通知消息可使用两种服务中的任何一种服务。订阅机制可使用快速分发服务，见第6章；设备复位机制应使用快速分发服务，见第12章。

注：对于某些类型的网络，可靠服务和快速服务可能相同。

5 设备模型

5.1 设备的可配置性

符合开放式控制架构的设备的可配置性可分为固定，可插拔，部分可配置和完全可配置，见表1。

表1 可配置性

可配置性	类型	描述
固定	静态	设备在固件编程时就永久地指配对象库和信号流拓扑
可插拔	静态	当设备离线时，通过插入和拔出硬件模块，调整物理控制，重新加载或重新调整软件或通过其他手动方法，可改变设备的对象库和信号流拓扑
部分可配置	动态	设备在线时，控制器可改变设备的信号流拓扑
完全可配置	动态	在“部分可配置”的基础上，设备在线时，控制器还可创建和删除设备内部对象

固定和可插拔设备称为静态设备，因为控制器不能改变其配置项。部分可配置和完全可配置设备称为动态设备，因为控制器可在线改变其配置项。

配置管理的详细内容见5.4.3和GY/T 322.2—2019。创建和删除对象见5.9和5.10。

5.2 对象寻址

在符合开放式控制架构的设备中，每个对象（即特定类的每个实例化）应由对象编号进行唯一标识。根据设备的可配置性，对象编号应在不同时刻进行分配，见表2。

注：在特定实现中，开发人员可使用特定的对象编号方案来优化设备性能和/或简化对象管理。例如，开发者可选择设备的对象编号值与内部表索引值相对应，以便于对输入命令的快速解码。

表2　对象编号分配

可配置性	ONo 分配时间刻
固定	设备制造时
可插拔	设备启动时
部分可配置	设备制造时
完全可配置	对象创建时

在完全可配置的设备中，当对象被删除并重新创建，且设备没有复位时，对象编号值不能被重新使用。极端情况下如果对象编号地址空间耗尽，对象编号值宜从最初值重新开始。

开发设备的人员可以根据需要选择对象编号值，但应遵守以下规则：

a) 对象编号应为 32 位整数。

b) 每个对象应有唯一的对象编号。

c) 对象编号不应为 0。

d) 从 01 到 4095 的对象编号值应给管理单元（见 5.6）和其他需要预定义对象编号的对象保留。它们应保留用于开放式控制架构的标准化。

e) 不同的对象编号不能指向同一个对象。

注：与一些其他媒体控制协议不同，ONo 不是基于某种分层级的多字段（i•j•k...）值。使用简单的 32 位 ONo 值是一种设计选择，旨在最大限度地提高协议效率，并最大限度地减少解析对象引用、处理命令响应和管理错误时的设备开销。

5.3　设备模型

5.3.1　概述

图1是开放式控制架构的设备模型。加框的元素表示对象。这些对象的类的定义见GY/T 322.2—2019。5.3描述可用类，举出示例和特有的结构布局。

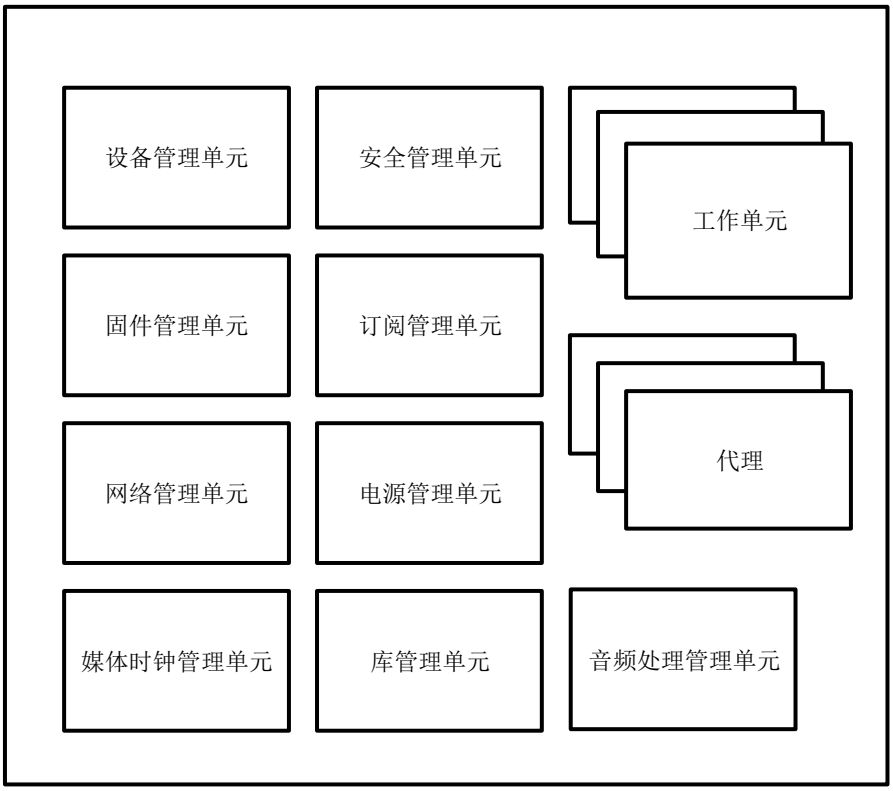


图1　开放式控制架构的设备模型

5.3.2　管理单元、工作单元和代理

开放式控制架构的设备模型包含如下三类对象：

a) 管理单元：该对象应是控制对象，它影响或报告设备的基本属性和总体状态。在每个设备中，每种管理单元类只应有一个实例（即一个设备管理单元，一个安全管理单元等）。每个管理单元应有一个标准对象编号。部分管理单元是必需的，其他是可选的，见 5.6 和 GY/T 322.2—2019。

b) 工作单元：该对象应直接控制设备的应用功能，见 5.4。包括：音频静音开关，增益控制，均衡器，电平监测，过载监测器，视频摄像机控制，信号属性，图像处理参数和信号处理功能等。工作单元应按表 3 分类。

表3　工作单元类型

类型	功能
执行器	控制应用功能(例如：开关)
传感器	检测信号参数和其他数值，并将其报告给控制器
块及块工厂	允许将相关对象组合成集合
矩阵	允许将对象集合作为二维数组来寻址处理
网络	描述数字网络连接特性

c) 代理：该对象应提供对同一设备中工作单元的间接控制。代理不应映射信号处理功能，但可影响一个或多个相关联的工作单元对象的信号处理参数，或提供其他应用控制功能。代理的详细描述见 5.5。

示例：

名字为OcaGrouper的代理以类似于模拟系统中压控放大器分组的方式实现控制参数的复杂分组。

这三类对象中所有类的详细定义见GY/T 322.2—2019。

5.4 工作单元类

5.4.1 执行器

执行器应控制设备内的信号处理和常规处理功能。执行器类见GY/T 322.2—2019。示例参见附录A。

在任何设备中，任何执行器类可根据控制应用功能的需要进行多次实例化。

执行器示例见表4。

表4 执行器示例

执行器类	功能
OcaGain	控制增益的功能——参见附录 A
OcaMute	控制信号静音的功能
OcaSwitch	控制多掷开关
OcaFilterParametric	控制参数均衡器区段
OcaDelay	控制信号延时
OcaDynamics	控制限幅器、压缩器、门控器或扩展器
OcaTemperatureActuator	控制温度设置

5.4.2 传感器

传感器应检测一些参数值并回传给控制器。传感器类在GY/T 322.2—2019中定义。传感器示例见表5。

表5 传感器示例

传感器类	功能
OcaLevelSensor	检测信号电平
OcaAudioLevelSensor	用标准的 VU 表或 PPM 表的平均值和轨迹来检测音频信号电平
OcaTemperatureSensor	检测温度

传感器值可通过周期性或条件性（例如，当其超过限定的阈值时）的方式，使用OcaNumericObserver代理（参见5.5.5和6.3）自动传送。

5.4.3 块

5.4.3.1 概述

块应为一种特定类型的工作单元，它可包含工作单元对象（包括其他块对象）和/或代理对象。块也可描述其所包含的工作单元之间的信号流拓扑。标准块类的定义见GY/T 322.2—2019。

块内的对象称为该块的成员。包含对象的块称为该对象的容器。块可被任意深度地嵌套。

任何工作单元和代理都应是一个块的成员。每个设备均至少应包含一个称为根块的块，它应当包含该设备的所有工作单元和代理。在简单设备中，所有的工作单元和代理均可属于根块。在复杂的设备中，工作单元和代理可部署到嵌套的块中。

管理单元对象不应属于块。

每个块都应用一个类（块类）描述。所有块类的根类应为OcaBlock。

注1：块是一个抽象容器，它对设备结构和控制流做了最小的假定。没有预先限制块可以包含什么样的块类型，也没有假设块可以包含什么样的对象。设备可以任意地设置块或者只有根块。

注2：块的预期用途包括：

a) 提供枚举功能（见 5.4.3.2）；

b) 存储信号流信息（见 5.4.3.6.4）；

c) 支持库功能（见 5.5.6）；

d) 对于可重配置的设备，允许创建及删除对象和信号路径，并提供创建和删除聚合处理功能的组织机制（见 5.9）。

5.4.3.2 块枚举

块所包含的工作单元和代理的集合称为块的对象集。对象集的成员列表称为对象列表。

OcaBlock应提供检索任何指定块的对象列表的方法。还应为设备提供选项，仅枚举直接包含的对象，或递归枚举所有直接包含的对象以及任意级别嵌套块中的所有对象。

注：为了发现设备中的每个工作单元对象，控制器只需要请求根块的递归枚举。

5.4.3.3 块管理

完全可配置的设备可允许创建、修改和删除块。开放式控制架构定义了控制器可使用的实现这些功能的方法。

删除块应删除它包含的所有对象。

对象和块创建见5.9。

5.4.3.4 块和对象寻址

对象寻址应严格按照对象编号进行处理。

注1：开放式控制架构未定义基于块包含的分层对象寻址方案。

注2：如果需要，制造商可以自由选择用作表示设备块结构的对象编号值。这种选择超出了开放式控制架构的范围。

5.4.3.5 块和控制聚合

块不应聚合控制功能，如联动、分组或主控。

注：开放式控制架构的控制功能的聚合由OcaGrouper类提供。OcaGrouper的机制与块边界无关，并且完全支持部分重叠分组功能，见5.5.3。

5.4.3.6 信号流

5.4.3.6.1 端口

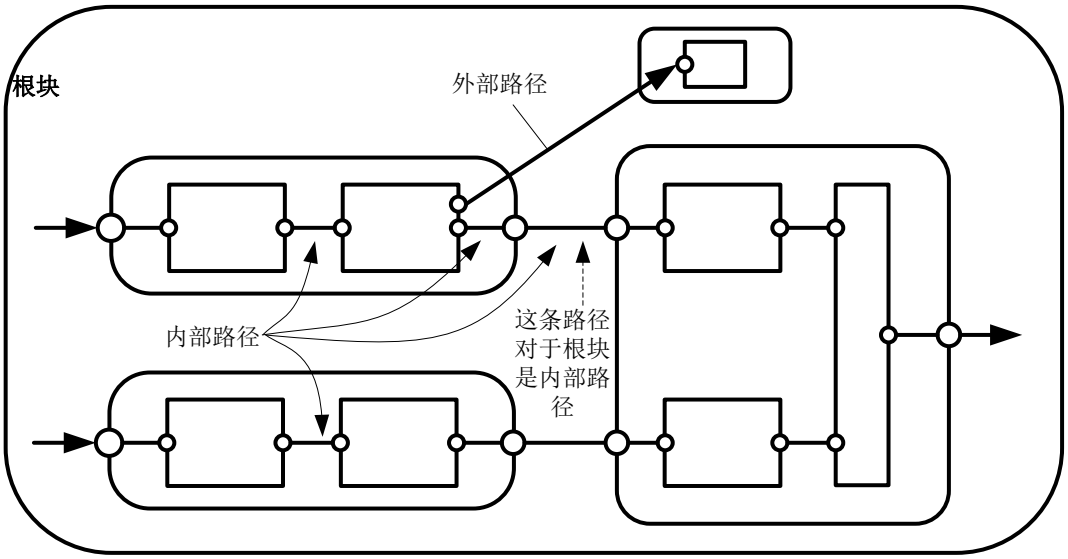
每个工作单元对象可以包含一个或多个端口用于定义信号流。端口是工作单元在实现处理功能时用来描述一个输入或输出信号通道的数据元素。

输入端口应表示信号流进处理功能；输出端口应表示信号流出处理功能。

注：本部分用代表信号处理功能的对象来描述那些在信号处理功能之间的信号连接。例如，为了表示从对象A描述的处理功能到对象B描述的处理功能的信号连接，文中将写为“从对象A到对象B的信号连接”。

5.4.3.6.2 块端口

块是工作单元，因此可有端口。这样的端口被称为块端口。块端口应作为中介端口，它的存在是用于定义块内对象之间与块外对象之间的信号连接点，如图2所示。



注：在本部分的图中小圆圈表示对象的端口，大圆圈表示块的端口，连线表示信号路径，圆角框表示块，直角框表示对象。

图2 信号流

块的输入端口应是信号进入块的连接点。对于块内的对象而言，块的输入端口应表现为一个信号源。块输出端口应是离开块的信号的连接点。对于块内的对象而言，块的输出端口应表现为一个信号接收器。

内部信号路径是同一块中两个对象之间的信号路径。外部信号路径是在不同块内的两个对象之间的信号路径，如图2所示。

5.4.3.6.3 信号路径

信号路径应表示在同一设备内特定输出端口和特定输入端口之间的连接。信号路径的输入和输出端口可以在不同的对象中或在同一对象中。不同的块中的对象之间的信号路径可包括或不包括块端口。

注：虽然不强制要求使用块端口，但是鼓励使用块端口。开放式控制架构允许直接连接不同的块中的对象。

5.4.3.6.4 块信号流

块的信号流应包括至少有一端在块内的所有信号路径的集合。信号流中的信号路径列表被称为信号流列表。

块的信号流应不包含从块端口延伸到块外部其他端口的信号路径。这样的路径属于整体包含它们的块。在简单情况下，这个块是根块。

一个设备与另一个设备之间的媒体传输连接不应被视为设备信号流的一部分。设备之间的媒体连接见7.2。

OcaBlock应提供检索任何指定块的信号流列表的方法。应提供如下选项：仅枚举直接包含的信号路径，或递归枚举所有直接包含的信号路径以及任意级别嵌套块中的所有信号路径。

图2表示一个典型的信号流。

5.4.3.7 示例

块和信号流示例参见附录B。

5.4.3.8 块工厂

在完全可配置的设备中，控制器应能够构建块，并用对象组装块，然后在这些对象之间连接信号路径。这些基本机制由块工厂来完成。

块工厂应是一个工作单元，它的工作是构建完全组装好的和“连好线的”块。每个块工厂应当是OcaBlockFactory类的一个实例，它已经被配置为用预定义的一组对象和信号路径来构造一种特定类型的块。

注：当控制器必须创建多个同一类型的块时，块工厂是有用的。

块工厂应是其构建每个块时实现的对象原型和信号流原型的容器。

开放式控制架构提供了以下两种创建块工厂的方法：

a) 设备制造时或可插拔设备启动时，可定义一个或多个块工厂。在这种情况下，这些块工厂应向控制器提供预先确定好的可实例化的块类型指令。

b) 控制器可以创建块工厂，即使用开放式控制架构的特定命令来定义它们将创建的块的配置。在这种情况下，控制器应自由定义随后将被实例化的新块类型。

根据设备类型，可实现这两种创建块工厂的方法之一或全部实现。

5.4.3.9 不用块工厂创建块

控制器可以通过实例化OcaBlock类的对象而无需使用块工厂来创建块，这时要向设备发送命令来创建和连接这个新块内的特定对象。OcaBlock类应提供必要的方法。

5.4.4 矩阵

5.4.4.1 概述

开放式控制架构涉及的矩阵应是交叉节点式音频矩阵概念的泛化。开放式控制架构的矩阵是相同工作单元的矩形阵列，这些工作单元分别在行和列上共享一个或多个公共输入和输出总线。

5.4.4.2 矩阵寻址

矩阵元素应通过坐标寻址。开放式控制架构使用(x，y)坐标，其中x是水平（列）编号，y是垂直（行）编号，如图3所示。

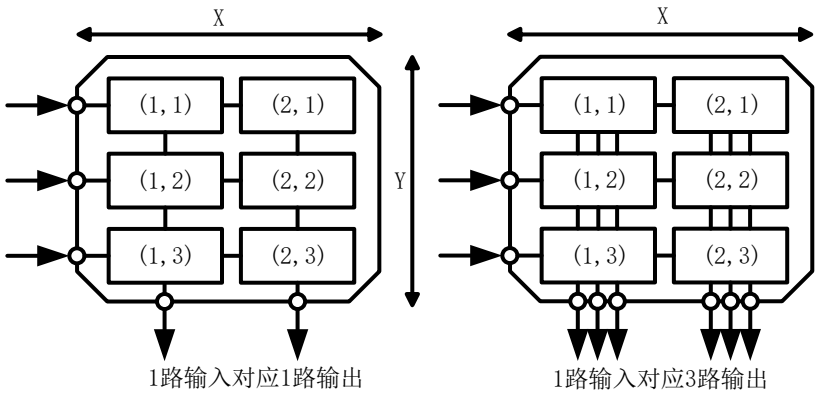


图3　开放式控制架构的矩阵

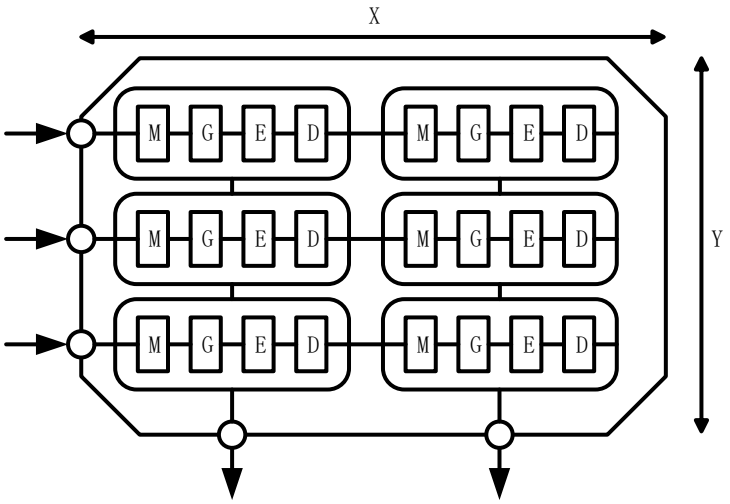
坐标值的范围应从1到行数或列数。

特殊值0应用于表示整个行或列。因此，例如，(0,2)表示整个第二行。此特点的使用将在5.4.4.5中展示。

5.4.4.3　复杂矩阵

如果矩阵中的工作单元是开关，则矩阵可以表示为常规的开关矩阵。如果工作单元是增益控制，则矩阵可以表示常规混音矩阵。在开放式控制架构中，矩阵的工作单元可以是任何类，甚至是块。

例如,图4表示一个块矩阵，其中每个块如图5所示。



注：图中“M”表示静音、“G”表示增益、“E”表示均衡器、“D”表示延时。

图4　复杂元素组成的矩阵

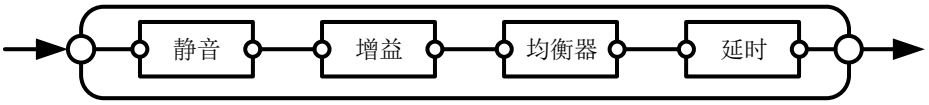


图5　复杂矩阵元素

5.4.4.4　矩阵结构

矩阵应是OcaMatrix类的实例，但是该实例不应包含全部或部分构成矩阵元素的工作单元。矩阵元素应分别实例化，并被标识为矩阵的成员。它们的类被称为该矩阵的成员类。简而言之，矩阵应收集其成员，而不包含它们。

矩阵的所有成员应为同一类，且应与矩阵在同一设备中实例化。

除了矩阵及其成员之外，矩阵还应包含一个称为矩阵代理的成员类的附加实例。控制器应使用矩阵代理通过坐标来访问矩阵成员的设置。

完整的矩阵如图6所示。

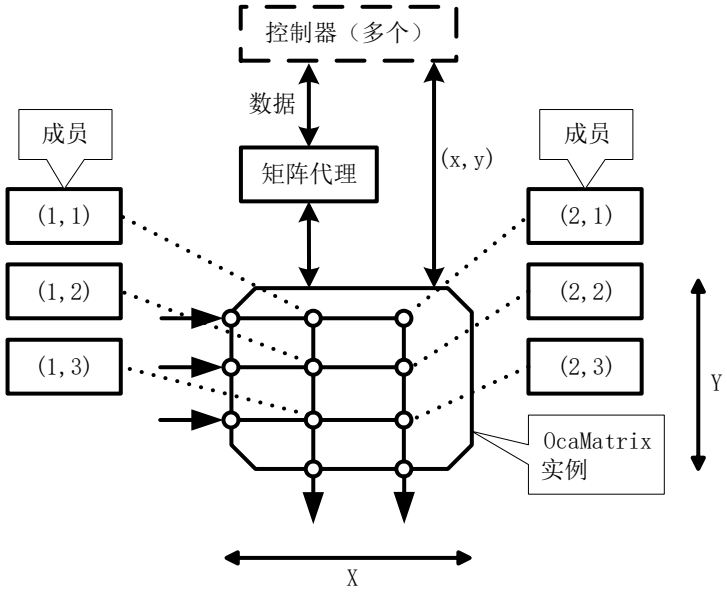


图6　矩阵结构

5.4.4.5　访问矩阵元素

控制器应使用SetCurrentXY(x,y)和SetCurrentXYLock(x,y)这两个方法来访问(x,y)坐标的矩阵成员：

a)　控制器应调用 OcaMatrix 的 SetCurrentXY(x,y)或者 SetCurrentXYLock(x,y)方法，指定目标坐标集。如果 SetCurrentXYLock(x,y)被调用，OcaMatrix 应给所寻址到的矩阵元素上锁。

b)　控制器应为所需的属性调用一个或多个矩阵代理的 Get 或 Set 方法。如果调用 Get，则应返回属性的当前值。如果调用 Set，则应设置属性的新值。

如果在步骤a)中SetCurrentXYLock(x，y)被调用，则控制器应调用OcaMatrix的Unlock(x，y)方法来解锁所寻址到的矩阵元素。

特殊值x=0和y=0可用于指定聚合集合操作，见表6。

表6　SetCurrentXY(x，y)示例

示例	功能
SetCurrentXY(x,y)	将第 y 行的第 x 列中的所有对象设置新值
SetCurrentXY(0,y)	将第 y 行的所有对象设置新值
SetCurrentXY(x,0)	将第 x 列的所有对象设置新值
SetCurrentXY(0,0)	将整个矩阵中的所有对象设置新值

矩阵的Get方法可以使用x=0检索整行，或者使用y=0检索整列。

当x=0并且y=0时，矩阵的Get方法不应使用。

另一种选择是，可以使用对象编号直接访问矩阵成员。这种情况绕过了矩阵机制，但不应损害矩阵机制。

开放式控制架构支持使用和管理所有对象属性的最大值和最小值。如果尝试进行的一个聚合的集合设置操作将导致任何成员的属性值超过其允许范围，该操作应被拒绝并显示错误提示。在这种情况下，所有成员属性不会被更改。

注：属性的最大值和最小值在对象创建时指定，这可能是在制造时，也可能是在设备硬件启动时或在操作时，它取决于设备的可配置性（见5.1）。

5.4.4.6　矩阵信号流

矩阵应具有与其成员的输入和输出信号端口分离的输入和输出信号端口（矩阵端口）。矩阵行应对应于矩阵输入端口；矩阵列应对应于矩阵输出端口。矩阵每行和每列均可有一个或多个端口。

在任何给定矩阵中，矩阵所有行输入端口的数量N_{in}对于所有行都应相同，矩阵所有列输出端口的数量N_{out}也均应相同。

矩阵成员的输入和输出端口应按以下规则顺序连接到矩阵行和列端口：

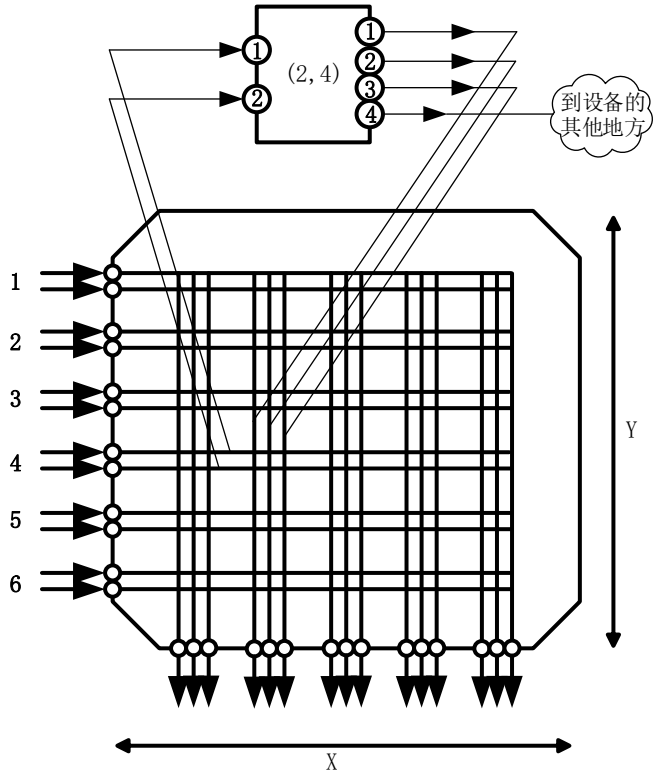
a)　行 y 中的成员的第 i 个输入端口应连接到矩阵端口 $i + N_{in}(y-1)$；

b)　列 x 中的成员的第 j 个输出端口应连接到矩阵端口 $j + N_{out}(x-1)$。

矩阵的成员宜有足够数量的端口来支持上述规则。具体来说，每个成员应该具有至少N_{in}个输入端口和N_{out}个输出端口。

成员可有若干不连接到矩阵端口的附加端口。这些附加端口的数量可以随不同成员而变化。

有关矩阵端口关系如图7所示。



每行端口数为2，每列端口数为3

图7　矩阵端口关系

5.4.4.7　部署

矩阵及其代理应与其成员在同一设备中进行实例化。对于能够跨设备的控制聚合函数，见5.5.3的OcaGrouper。

5.4.4.8　应用注释

5.4.4.8.1　非混音应用

矩阵的定义十分通用，它不仅能用于表示传统的音频混音矩阵，还能用于需要将对象集合寻址为一维或二维数组的其他应用。这样的应用可使用或不用矩阵输入和输出端口。

示例：

扬声器分频器可以被表示为分频器通道的矩阵，其中每个通道是包含通用的滤波器、增益单元、延时和动态控制的块。在这种情况下，这些通道可以共享一个公共的输入端，这个输入端可以用一个矩阵的行端口来表示。然而，它们将使用单独的输出端口，因此不需要矩阵输出端口。

5.4.4.8.2　非累加方式的输出聚合

当多个成员输出连接到矩阵的某单个列端口时，成员信号应累加以产生最终的列信号输出。然而，开放式控制架构的控制概念不要求这样做；矩阵对象可用于控制其他类型的输出聚合算法。

5.5 代理类

5.5.1 概述

本部分定义了代理类的概念和语义。具体的代理类属性、方法和事件见GY/T 322.2—2019。

5.5.2 OcaNetwork 和 OcaStreamNetwork

这些代理类见第7章。

5.5.3 OcaGrouper

5.5.3.1 概述

OcaGrouper应定义一个对象（编组器），它以属性值相关联的方式使其可以作为单个值进行控制。

注 1：编组器支持音频控制功能，如各种已知的绑定、联动、主控、分控和 VCA 控制。

编组器不应有信号通过，它们只应影响控制参数。

OcaGrouper是其他编组器的根类，其他的编组器类应在此基础上定义。在下文中，编组器表示OcaGrouper类的实例或OcaGrouper子类的实例。

执行器的编组器应通过单一输入参数对多个执行器对象提供控制。

传感器的编组器应准许用单一输出值对多个传感器对象进行观测。传感器的编组器将在开放式控制架构的后续版本中定义。

注 2：开放式控制架构的当前版本只定义了执行器的编组器。

5.5.3.2 组

一个编组器应包含一个或多个组。一个组应是相同类的工作单元或代理的集合。编组器中的所有组应收集同一个类的对象。

组应聚合对象的所有属性，而不是个别属性。这种聚合应依旧保持这些对象不同属性之间的区别。

组设定值是指用于特定属性的组设置。

示例：

GY/T 322.2—2019中定义的OcaFilterParametric类表示参数均衡器，它具有三个主要属性-频率，Q值和带通增益。在一组OcaFilterParametric对象中，所有这些属性都应单独分组。因此，该组应为每一个频率，Q值和带通增益维护单独的组设定值。

组设定值应通过5.5.3.4中描述的组代理机制实现。

5.5.3.3 重叠的组成员关系

在一个正在运行的媒体系统中，同一对象可分属多个组。

示例：

在多分频扬声器的立体声增强系统中，左声道低音扬声器功率放大器的增益控制对象可以通过主增益组，左侧增益组和低音扬声器增益组来控制。

本部分中，将共享了一个或多个对象的组称为重叠组。

对象是两个或多个组的成员时，该对象的属性的设定值将取决于该对象属于的所有组中相应组设定值的累积作用的结果。

注1：由于所有属性都有取值范围限制，所以重叠组设定值的累积作用可能会导致属性超出取值范围。必须防止或至少管理累计作用。为了管理取值范围限制，编组器必须知道所有重叠的组以及哪些对象属于哪些组，并且必须具有按照应用所要求的方式来处理取值范围限制相关问题的机制。这就是 OcaGrouper 被赋予了包含多个组能力的主要原因。

注2：为了使编组器的取值范围限制管理机制能在某个特定的媒体网络中工作，网络宜配置该为所有重叠的组共用一个公共的编组器。这是一个应用指南。

5.5.3.4 组结构

5.5.3.4.1 概述

编组器应能够管理非重叠和重叠组的取值范围限制问题。

组结构中常用表7所示的术语。

表7 组结构术语

术语	描述
居民	编组器所知道的对象
组	编组器所定义和操作的组
登记	将一个居民关联到一个组
成员	在组中登记的居民
组代理	控制器应用该对象来访问组设定值

编组器应提供以下功能：

a) 创建或删除组和组代理；

b) 在编组器中注册和注销居民；

c) 在组中登记或删除居民；

d) 当组设定值改变时，能计算并设置新的属性值；

e) 以某种方式管理超过范围和低于范围的组设定值，使得要被设置的居民不会超范围地设置它们的属性值；

f) 处理编组器和居民之间连接丢失时出现的错误状况。

同一编组器的所有居民都应是同一个类的实例。编组器应对其居民进行引用收集，但不得创建，销毁或包含这些居民。编组器的居民可以驻留在网络的任何地方，它们不需要与编组器在同一设备中被实例化。

注：将编组器视为某种类型的数据矩阵是有帮助的。矩阵的行是组，列是居民，其中每个交叉点包含了与该组（行）中这个居民（列）成员身份相关的信息。

编组器有主从模式和/或对等模式两种操作模式，且至少应支持这两种模式中的一种。编组器的模式由其Mode属性值决定。

5.5.3.4.2 主从模式

在主从模式下，每组中的参数值应通过组代理访问。要更改组设定值，控制器应更改组代理中相应的属性值。

每一个组均应有一个组代理。这个组代理的类应与其组成员的类相同。当组被创建和删除时，相应的组代理也应由编组器自动实例化和删除。

组代理应与编组器驻留在同一设备中。

5.5.3.4.3 对等模式

在对等模式下，不应创建或使用组代理。只要任何一个成员的设定值改变时，这个组设定值就应该被改变。所有的组成员都应像组代理一样起作用。

5.5.3.4.4 简单示例

典型的用于控制三分频扬声器系统立体声设置的编组器配置如图8所示。

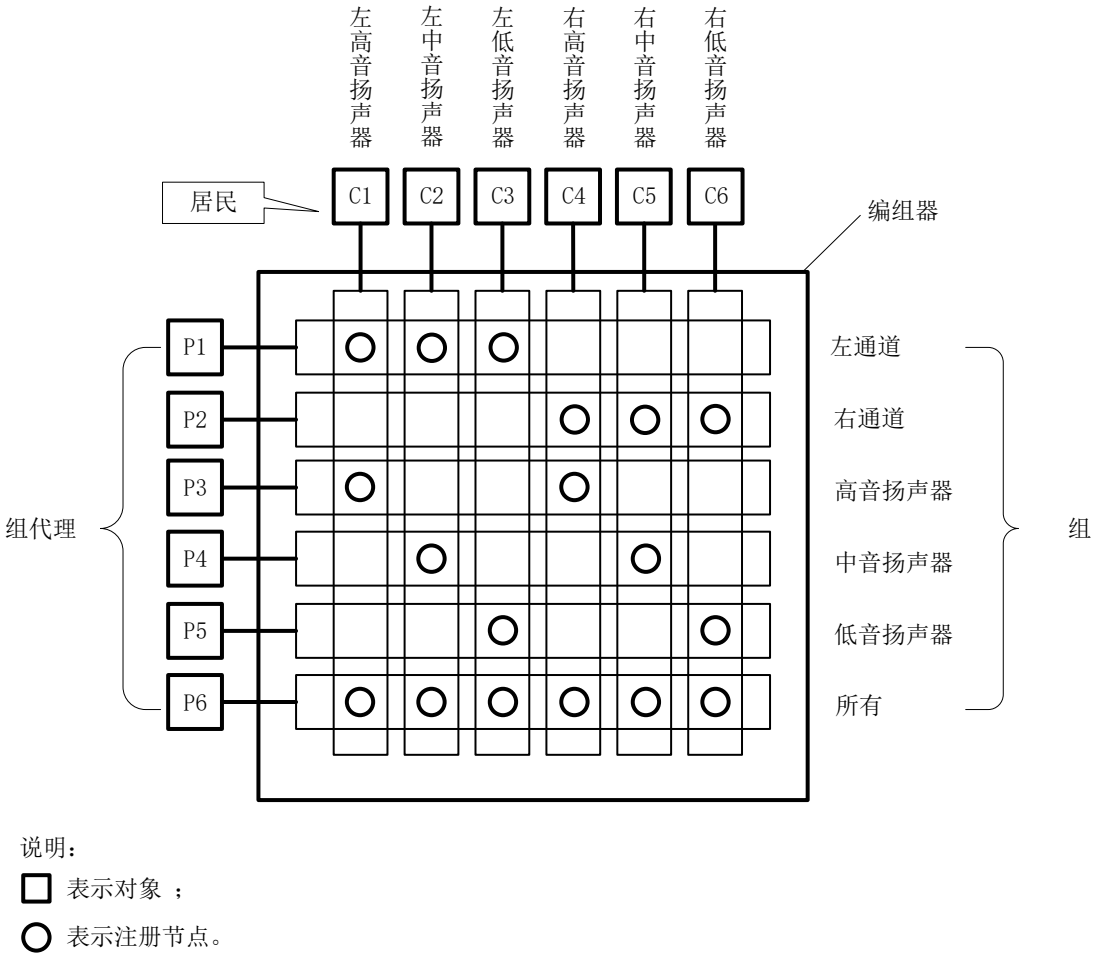


图8 典型编组器

5.5.3.5 聚合和饱和的规则

5.5.3.5.1 概述

居民的设定值应由聚合规则确定,这个规则决定了计算设定值所使用的算法。组应由饱和规则所管理,这个规则确定了超范围的情形该被如何处理。这些规则是随不同的居民类、属性类型以及产品而变化。

5.5.3.5.2 聚合规则的选项

编组器可使用多种算法来计算个体居民的设定值。聚合规则应取决于被分组属性的数据类型、所涉及的设备以及应用。

SUM规则可用于连续值的参数，例如增益和延时：

示例1：

$$N=sum(N_1、N_2……N_k)+J$$

式中：

N ——居民的设定值；

N_1、N_2……N_k ——适用组的设定值；

J ——偏移值。

“适用组”指该居民所隶属的组。偏移值是每个居民的私有值，这使得其设定值不同于组的设定值。这个私有值可以被认为是一个居民特有的“微调”设置。

在主从模式下，偏移值直接由控制器调用个体居民的Set（...）方法的结果产生，它绕过了编组器。在对等模式下，不应发生此行为。

LINK规则可用于离散值参数的联动，例如选择器开关设置。

居民的设定值等于适用组中最新改变的设定值。

对于静音这样的参数，AND，OR和XOR等这样的布尔型规则可能是除了LINK缺省规则之外有用的规则。

示例2：

$$N=XOR(N_1、N_2……N_k)$$

当具有多个属性的对象被分组（例如，参数均衡器）时，这些属性所用的聚合规则可各不相同的。

5.5.3.5.3 饱和规则选项

组应使用下列两个基本的饱和规则之一：

a) 饱和：编组器应允许所有设定值的变化，但应将舍尾值发送给个体居民，以避免超出范围条件。当组设定值返回较多的中间值时，应恢复聚合规则的正常操作。

b) 非饱和：编组器应拒绝使任何居民属性值超出范围的任何设定值的修改动作。

5.5.3.5.4 连接失效

当编组器与一个或多个居民连接失效时，可能需要特殊的处理以保持系统的完整性。这一点在组的设定值被修改了一段时间后，如果失连的居民再次与组重新连接时显得尤其重要。

OcaGrouper应定义一个名为StatusChange的事件，使得控制器可订阅该事件得以获得居民连接失效的消息。

5.5.3.5.5 标准 OcaGrouper 类规则

标准OcaGrouper类应定义缺省饱和、聚合和连接失效规则：

a) 缺省的聚合规则应取决于被分组属性的基本数据类型。基本数据类型相当于它的底层机器数据类型。缺省值见表 8。缺省饱和规则应为非饱和。

表8 缺省聚合规则

基本数据类型	聚合规则
Float	SUM
Integer	LINK
Bool	LINK
Enum	LINK
String	LINK
Others	（未分组）

b) 缺省连接失效行为应如下：连接失效后，标准 OcaGrouper 对象应继续控制任何与之保持连接的居民；当连接恢复时，编组器应更新失连居民的属性值。只要这些失连居民的设置值在其失连时间内没有改变，这种更新将恢复设定值的前后一致性。

注1：OcaGrouper 的各种子类被设计实现以满足一系列应用的需求。不同的实现方案将要求不同的连接失效处理行为。

注2：编组器可在普通设备、专用符合开放式控制架构的控制服务器或系统控制器中实例化。产品或系统的设计时可选择在哪里部署编组器。

5.5.4 OcaRamper

OcaRamper应定义一种机制（淡入淡出单元），控制器可通过该机制使设备自动执行增量或预先安排的参数变更。

每个淡入淡出单元均应绑定到一个特定的工作单元属性。淡入淡出单元参数应包括目标属性值，斜坡持续时间，斜坡起始时间和斜坡插值的规则。

每个淡入淡出单元均应保持一个状态属性，其值可以由控制器监视，以确定斜坡状态，例如：等待启动，正在进行，完成或中止。

注：使用OcaRamper的基本原理理由如下：由于网络性能的原因，通过在网络上发送一系列指令命令序列来实现增量更改（淡入，淡出和交叉淡入淡出）通常是不切实际的，因为这种方式需要过多的控制流量。如果不受制于网络延时影响，斜坡变化动操作的定时精度将会更精确。此外，在控制器中可能无法实现预设的参数改变，因为当控制器不运行时应用系统有可能需要执行此类更改。

5.5.5 OcaNumericObserver 和 OcaNumericObserverList

OcaNumericObserver应定义一个观测对象，用来监视另一个对象中指定的数字型属性的值，并在某些条件下将该数值通知控制器。OcaNumericObserver是事件和订阅机制的一部分，见6.3。

OcaNumericObserverList 应定义一个和 OcaNumericObserver 类似的观测对象。OcaNumericObserverList应能监视由给定列表中列出的位于其他对象中的数字型属性的值。

5.5.6 OcaLibrary

5.5.6.1 参数集合

5.5.6.1.1 概述

在这一领域开放式控制架构的特征是围绕着块机制组织（见5.4.3）。库特征是可选的，设备不必实现任何库相关的对象或功能。

OcaLibrary应定义一种机制来处理预存在设备中的参数值集合。

注：这种参数集合曾被称为“预设”，“Patch”，“存储”或“场景”。

开放式控制架构定义了参数集和Patch两种预存储的参数集合。

5.5.6.1.2 参数集

参数集应是块中特定类预存的一组属性值。只读的属性（例如，类ID）不应由此机制保存或恢复。

将这些值安装到块实例中的行为称为将参数集应用于该块。将特定的参数集应用于特定块实例的存储定义称为参数集作业。

不应要求参数集包含其目标块中的所有属性值。参数集可只包含一个值。将参数集应用于块应仅改变那些有被该参数集包含的属性值，其他的属性应保持不变。

定义参数集的块类称为该参数集的目标块类。参数集可应用于任何数量的目标块类的实例。

参数集的二进制格式是设备相关的，没有在开放式控制架构中规定。在开放式控制架构中，参数集应被视为二进制大对象。开放式控制架构包含了上传，下载，创建和管理参数集的原语，但这些原语不包括用来检查这些参数集的函数。

示例：

假设调音台定义了一个名为InChannel的块类来表示输入通道。如此，具有32个输入的调音台将具有该类的32个实例。这个调音台可以为InChannel定义一个或多个的通道参数集。这些参数集中的任何一个都可以应用于InChannel的任何实例。

5.5.6.1.3 Patch

Patch应由参数集作业的集合组成。执行Patch包中所有的作业操作的行为称为Patch应用于设备。Patch应用于设备仅影响该Patch参数集所包含的参数值，其他参数应保持不受影响。

注：术语“Patch”继承自MIDI设备控制协议中的类似概念。MIDI“Patch”是将设备配置为特定状态的一组MIDI命令。

5.5.6.2 OcaLibrary 结构

参数集的集合称为参数集库。Patch的集合称为Patch库。设备可实现的参数集库或Patch库的数量是无限的。

注：Patch与参数集库不同，因为参数集库是参数集的集合，而Patch是参数集作业的集合。

设备中的每个库应由OcaLibrary的实例表示。所有参数集库和Patch库的实例，都应由OcaLibraryManager对象收集。

5.5.6.3 创建参数集

开放式控制架构定义了两种在设备中创建并存储参数集的方法：

a) 控制器可将参数集下载到设备的一个指定库中；
b) 控制器可请求设备中的OcaBlock对象将其所有属性值作为参数集保存在该设备的某个特定库中。此“快照”操作应捕获块内所有对象的所有属性的值，并将它们作为一个参数集保存在指定的库中。

5.5.7 OcaMediaClock

OcaMediaClock应描述设备使用的特定内部或外部媒体时钟。它应包括时钟源，频率，速率和锁定状态这些指定参数。

设备可定义任意数量的媒体时钟;每个媒体时钟都应由OcaMediaClock实例表示。所有这些实例都应被媒体时钟管理单元(OcaMediaClockManager类)收集。

如果设备没有网络可控的时钟特性，则不需要实例化OcaMediaClockManager或OcaMediaClock。

5.5.8 OcaEventHandler

见第 6 章。

5.6 管理单元类

管理单元类见表9。

表9 管理单元类

对象编号	类名	功能
1	OcaDeviceManager	管理与整个设备相关的信息-包括型号和序列号，设备名称和角色，整体工作状态和设备更新锁
2	OcaSecurityManager	管理安全密钥
3	OcaFirmwareManager	执行固件更新
4	OcaSubscriptionManager	管理订阅，设备用来通知控制器重大件的结构。见第 6 章
5	OcaPowerManager	管理设备电源状态，包括多个电源供电、电池供电
6	OcaNetworkManager	管理设备所连接的控制和媒体传输网络
7	OcaMediaClockManager	管理媒体时钟的选择和控制
8	OcaLibraryManager	管理设备的 Patch 和预设置，处理预存的设备配置和参数设置的元素。见 5.5.6
9	OcaAudioProcessingManager	管理全局音频处理参数
10	OcaDeviceTimeManager	提供对设备日历时钟的访问
100	OcaBlock	根块

注：对所有设备来说部分管理单元对象是必需的，其他管理单元对象是可选的，见GY/T 322.2—2019。

5.7 标准对象编号

每个设备应给某特定对象分配标准对象编号。这些对象编号在表 9 中给出。

5.8 对象文本标识

每个开放式控制架构的对象均应包含一个名为 Role 的只读文本型属性,它应说明工作单元在设备中的作用。例如：“前置放大器增益（Preamp Gain）”。

此外，每个工作单元对象和代理对象应包含一个名为Label的可写文本型属性，控制器可以使用它来记录该对象在应用环境中的作用。例如：“埃尔维斯增益（Elvis Vocal Gain）”。

5.9 构造对象

5.9.1 概述

根据定义，完全可配置的设备应允许控制器构造工作单元和代理对象。这些功能应由OcaBlock的方法提供：

对于完全可配置设备，控制器可调用ConstructMember来构造对象。当被调用时，ConstructMember应根据特定对象的参数来构造该对象。这些参数称为构造参数，在GY/T 322.2—2019中为每个类定义了构造参数。当控制器调用ConstructMember时，它可能为部分或所有构造参数赋值。GY/T 322.2—2019为未指定值的参数提供了缺省值。

示例：

OcaSwitch类定义了一个通用的带标签的n掷开关。在构造时，控制器可以指定掷数及每掷的标签。缺省值是带有空白位置标签的双掷开关。

在完全可配置的设备中，控制器应能够构建块，用对象来填充它们，并在这些对象之间连接信号路径。应提供如下两个选项来完成这些工作：

a) 控制器可调用 ConstructMember 来构造块，然后再次调用 ConstructMember 在新块中构造对象。然后，控制器可按需调用 AddSignalPath 来定义块内的信号流。

b) 控制器可调用 ConstructMemberUsingFactory，引用一个特定块工厂对象，来构造由块工厂对象预定义了对象集和信号流的块。

5.9.2 块工厂

每个块工厂应是类OcaBlockFactory的一个实例，见GY/T 322.2—2019。块工厂应是构造一个特定类型的块的对象，且具有预定义对象集合和信号路径。块工厂应与其创建的块驻留在同一设备中。

创建块工厂应有如下三个选项：

a) 块工厂可由设备的固件定义；

b) 如果设备是可插拔的，则可在设备启用时定义块工厂；

c) 如果设备是完全可配置的，则控制器可使用开放式控制架构的命令来创建块工厂。

根据设备的可配置性，可实现以上三个选项中的部分或全部。

5.10 删除对象

部分设备中，对象可用其所在块的DeleteObject方法来删除：

a) 在静态的，可插拔的和部分可配置的设备中，不可删除对象；

b) 在完全可配置的设备中，可删除任何工作单元（包括块）或代理。

当删除工作单元时，连接到它的所有信号路径都应删除。当删除块时，块中的所有对象都应删除。

6 事件和订阅

6.1 订阅、事件、发送器和通知

6.1.1 概述

订阅表示两个对象之间的一种持续稳定的关系，当设备特定条件发生时，一个对象就会自动给另一个对象发送更新消息。这种条件称为事件，发送消息的对象称为发送器，发送的消息称为通知。每种事件都应定义特有的通知类型。发送通知的对象触发相应的事件。

一个发送器可实现多类事件，因此可发送多类通知。

6.1.2 可靠订阅或快速订阅

订阅应由订阅管理单元创建，以响应控制器对订阅管理单元中AddSubscription方法的调用。订阅用两种方式之一创建：可靠或快速。可靠订阅应通过可靠消息传递服务发送通知；快速订阅应通过快速消息传递服务发送通知。消息传递服务类的描述见4.3.2。

6.1.3 订阅删除

订阅应一直存在直到被弃用或被控制器删除。弃用订阅是订阅器不再在网络中出现的订阅。在大多数实现中，可用keep-alive消息（见11.2.1）检测订阅器故障。

当设备检测到订阅器故障时，它应删除故障订阅器产生的所有订阅。

6.1.4 订阅事件处理程序

接收通知的元素称为事件处理程序。事件处理程序是代理类OcaEventHandler的实例的OnEvent方法。拥有事件处理程序的对象称为订阅器。

当订阅事件发生时，发送器应向订阅器的事件处理程序发送一个通知。此通知等效调用订阅器的OnEvent方法。通知应包含允许事件处理程序执行适当处理的信息。

开放式控制架构不限制事件的订阅器数量，也不限制事件处理程序可能参与的订阅数量。

注：尽管开放式控制架构对订阅器和订阅的数量不限制，实际应用中不同设备仍有限制。

6.2 PropertyChanged 事件

根类OcaRoot应定义名为PropertyChanged的事件。当该事件应用于某对象时，对象的任一属性值变化都应触发该事件：

a) 监测信号引起的对象的状态变化；
b) 监测设备全部状态；
c) 更新控制器，匹配用户对前面板的调节；
d) 多控制器系统中，更新不同控制器之间参数状态。

当对象的任一属性发生变化时应触发PropertyChanged事件，返回给订阅器的数据应标识对象的哪些属性值发生了变化。

注：通过类继承机制，每个开放式控制架构的类都定义PropertyChanged事件。因此，控制器只需订阅PropertyChanged事件即可方便地监测一个对象的属性值。由于一个设备的所有可控制参数都驻留在属性中，因此PropertyChanged事件可以提供设备的所有可控操作状态的访问。

6.3 数值观察器的用法

OcaNumericObserver类在GY/T 322.2—2019中定义。每个OcaNumericObserver对象应监测一个特定参数值，只要达到观察门限，应触发名为observation的事件。

若需要，数值观察器可在设备中创建，指定相关的门限测试数值、周期重复率，或两者兼备，并将其与被观察的属性绑定。控制器就可订阅数值观察器的observation事件。随后，只要满足数值观察器的门限，数值观察器应将观察通知发送给控制器的事件处理程序。

注1：数值观察器可在设备制造时创建。对于动态设备，也可随后由控制器创建。

注2：数值观察器通常和传感器对象结合使用（例，音频电平传感器），但可一样用于与任何对象的任何属性。例如，可定义一个数值观察器，当功放增益提高至阈值电平以上时，数值观察器就发送通知。

示例：

用数值观察器实现控制器中信号指示灯的，如图9所示。

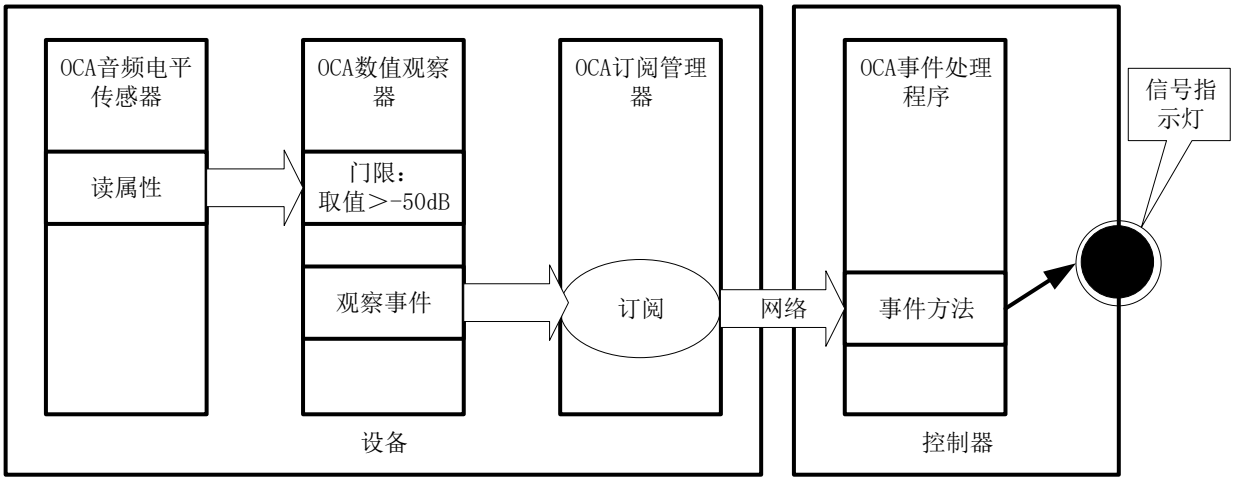


图9　用数值观察器实现信号指示灯

7　网络系统

7.1　概述

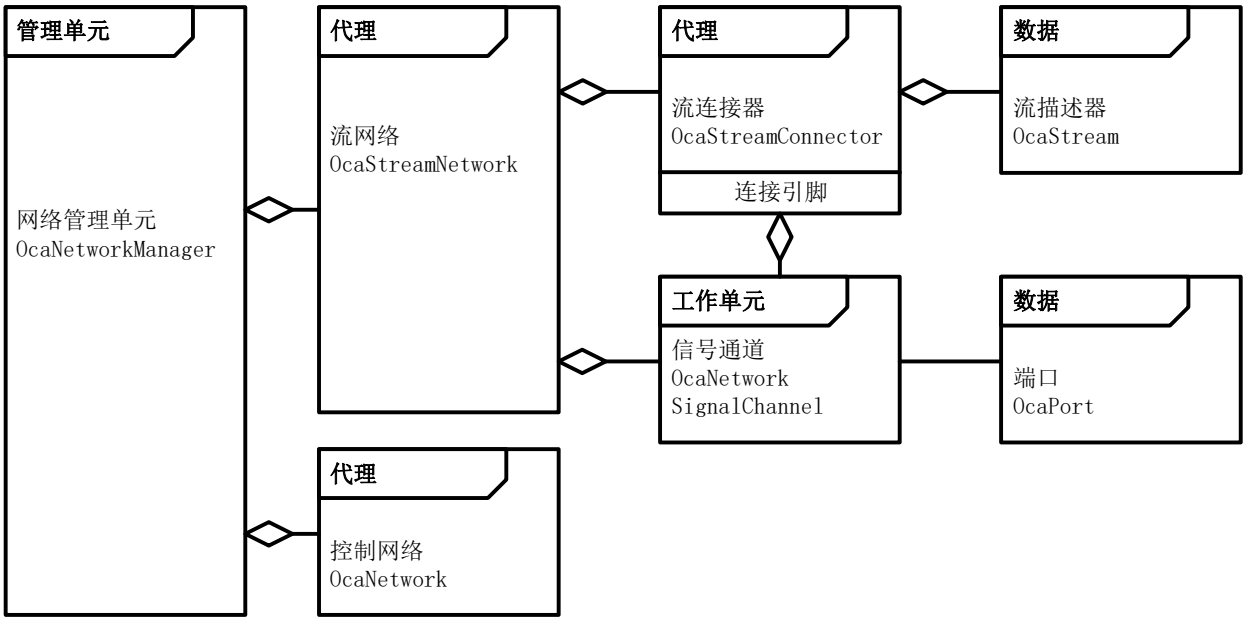
在开放式控制架构中，网络应概括为可支持监控，或流媒体传输，或同时支持两者的网络。对支持媒体传输的网络，控制接口必须是通用的，以便允许开放式控制架构与不同类型的媒体网络协同工作。

开放式控制架构应支持某一时刻属于多个网络的设备。多个网络可是同种类型或不同类型。

开放式控制架构中涉及网络的类见表10，这些类之间的关系如图10所示。

表10　网络类

类名	类型	功能
OcaNetworkManager	管理单元	收集设备所属的所有网络
OcaNetwork	代理	代表载有控制流量的网络
OcaStreamNetwork	代理	代表流媒体和可能载有控制流量的网络
OcaStreamConnector	代理	代表流入或流出媒体流的一个连接点
OcaStream	数据	代表一个流入或流出的媒体流
OcaNetworkSignalChannel	工作单元	允许一个外部输入或输出信号到设备内部的一个或多个信号通路的连接



说明：

表示聚合关系，平端对象被聚合到菱形端的对象中。

图10　网络对象关系

图10中表示了流网络（OcaStreamNetwork）对象和控制网络（OcaNetwork）对象。OcaStreamNetwork对象应能同时管理信号传输和开放式控制架构的控制流。OcaNetwork对象应能管理开放式控制架构的控制流。设备可使用两种类或其中之一：

a)　如果开放式控制架构用作管理媒体传输连接，应使用OcaStreamNetwork；

b)　仅需控制设备，不需媒体传输功能，可用OcaStreamNetwork或OcaNetwork；

c)　新设计宜用OcaStreamNetwork；

d)　如果一个设备属于多个网络，根据需要宜使用多个OcaStreamNetwork实例。

7.2 媒体传输连接管理

通过OcaStreamNetwork和其相关类（图10），开放式控制架构的控制器应能访问媒体网络连接管理功能，用于建立，监测和删除设备之间的媒体传输连接。

7.3 开放式控制架构适配

7.3.1 概述

开放式控制架构的网络类应构成管理各种媒体传输网络连接的基础。为了支持一些特定的媒体传输网络，还需要一些调整。开放式控制架构对特定网络类型的配置应称为开放式控制架构的适配。

开放式控制架构的适配应包含配置和使用开放式控制架构的网络类的规则，并且可附加定义开放式控制架构的标准类的特例（子类）。

有些适配需要对同一物理网络生成OcaStreamNetwork类的多个实例。

示例：

网络的一个开放式控制架构适配利用TCP/IP协议实现监控，但是用不同的协议套件进行媒体传输，需要两个OcaStreamNetwork代理，一个用于IP控制流量，另一个用于媒体流。

开放式控制架构的适配超出开放式控制架构的范围。

7.3.2 媒体网络的种类

开放式控制架构的媒体传输管理方案应支持基于流和基于通道的媒体网络：

a) 基于流的媒体网络，应是将一些信号通道汇聚成称为流的单向组的媒体网络。这种网络中，应用在流基础上进行设备间连接，不同流的通道数可不同。

b) 基于通道的媒体网络,应是一种应用在信号通道基础上进行设备间连接的媒体网络。在此网络中，信号通道可选择是否分组到流中，如果进行分组，则通常会被自动管理，并且对应用不可见。

开放式控制架构对媒体流格式、采样率、编码方式以及其他媒体数据表达并没有限制。

7.3.3 实例

基于流和基于通道的连接管理的实例参见附录C。

7.3.4 OcaStreamNetwork 类

开放式控制架构的媒体网络管理机制应围绕OcaStreamNetwork类组织。OcaStreamNetwork对象应存储基本网络访问信息，且应持有两个关键集合——流连接器和信号通道。

注：大部分设备仅需一个OcaStreamNetwork对象去处理所有媒体流和控制流。某一时刻设备属于多个网络，则需要用多个OcaStreamNetwork对象。

每个OcaStreamNetwork实例应为它代表的网络提供如下功能：

a) 启动、暂停和关闭网络接口的功能；

b) 标准化的网络状态指示；

c) 能够获取和设置主机名称或 ID，以便在网络中公告设备自身；

d) 识别网络硬件链接类型（TCP/IP 以太网，USB 等）；

e) 识别用于网络输入和输出的软件接口；

f) 识别网络使用的控制（如有）和媒体传输（如有）协议；

g) 传输性能和错误统计；

h) 关联类的聚合。

OcaStreamNetwork和它的关联类的特定使用应取决于媒体网络是基于流还是基于通道。

7.3.5 模式 1：基于流的媒体连接管理

7.3.5.1 概述

对基于流的连接，流应连接到流连接器，流内的通道应连接连接器的引脚，连接器引脚将连接到内部设备的信号路径。

7.3.5.2 流连接器

流连接器应是OcaStreamConnector类的实例。

流连接器应是代理对象，为了发送和接收流数据，这个对象绑定了与外部媒体传输流的设备。

OcaStreamConnector对象应包含允许控制器创建和删除媒体流连接的方法。OcaStreamConnector对象应能通知订阅控制器媒体流连接已建立或中断，以便响应外部设备和控制器的请求。

OcaStreamConnector对象应包含识别媒体传输连接安全的特性。媒体传输安全的具体实施不在开放式控制架构的范围之内。

流连接器可以是源（输出）连接器或目的（输入）连接器。不应支持双向连接器。源连接器（不是目的连接器）可连接多个输出流。

每个连接至流连接器的流，其连接参数应记录在流描述符中。见7.3.5.3。

7.3.5.3 流描述符

流描述器应是在OcaStream数据类型类中定义的一个数据元素。它应保持网络媒体流的属性，每个流连接器应收集其连接的每个流的描述符。因此，OcaStreamConnector类和OcaStream类有一个聚集关系，如图10所示。

一个源（输出）流连接器可连到多个流，因此可拥有多个流描述符。一个接收（输入）流连接器最多连到一个流，因此最多应拥有一个流描述符。

7.3.5.4 连接器引脚

流连接器类似于传统的多引脚信号连接器。继续这个比喻，每个流连接器应具有一个或多个连接器引脚的集合。每个连接器的引脚应是一个数据元素，它将单一信号从连接流映射到OcaNetworkSignalChannel对象。使用5.4.3.6描述的标准信号流机制，OcaNetworkSignalChannel对象的端口可依次与一个或多个设备内部的内部信号连接。

7.3.5.5 信号通道

信号通道应是OcaNetworkSignalChannel类的实例。

信号通道应是工作单元对象,它提供了一个特定的连接器引脚和信号路径或设备内部路径之间的绑定。每个信号通道应包含一个可连接设备内部信号路径的OcaPort属性。

信号通道可是源（输出）通道或目的（输入）通道，不应支持双向通道。

一个源（输出）信号通道可与多个连接器的多个连接器引脚关联，一个目的（输入）信号通道最多应与一个连接引脚关联。

7.3.6 模式 2：基于通道的媒体连接管理

对基于通道的连接，网络中的每个输入、输出信号通道应与设备的一个信号通道对象（7.3.5.5）连接。信号通道对象应是OcaNetworkSignalChannel类的实例。信号通道对象应存储与此通道连接的相关连接信息。

一个输出通道可与网络中的多个通道连接；一个输入通道应最多只能与网络中的一个通道连接。在内部，通过OcaPort每个信号通道对象按照开放式控制架构的通用信号流规则（5.4.3.6），可连接到设备间的信号路径。

应有一个OcaStreamNetwork对象描述网络。这个对象应聚合所有OcaNetworkSignalChannel对象。

这些关系如图11所示。

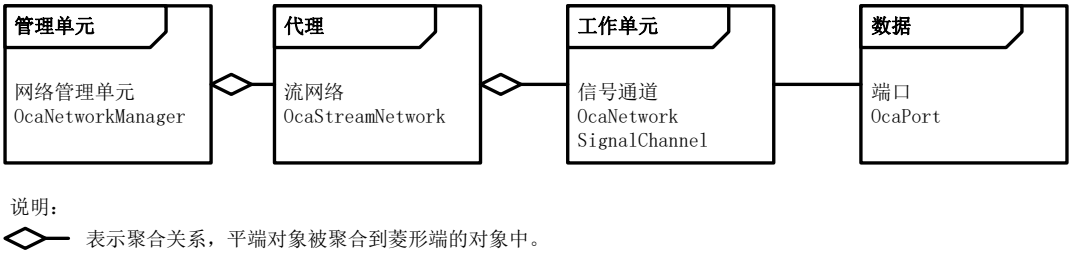


图11 基于通道的类

8 会话

开放式控制架构的协议是面向会话的，这意味着以下几点：

a） 请求和响应是相关对；

b） 对象应与其他开放式控制架构的对象具有永久应用关系；

c） 设备故障应及时发现；

d） 设备应能定期和持续地向订阅控制器报告参数变化（例，信号电平）；

e） 当传输中断发生时，控制器与设备的关系应以可预见的方式维持或被删除。

注：网络中可能包含几千个符合开放式控制架构的设备。用单一的中央控制器直接控制成百上千的设备，维护每个单独的传输连接，这通常不太实际。这种情况下，使用多控制器层次化结构可实现间接控制，伴随适合于应用的连续层级的聚合控制功能。聚合特征，主要是OcaGrouper类（见5.5.3）会帮助这种实现。

9 安全

开放式控制架构的目标是支持在足够安全级别下运行的网络应用，能满足：

a） 紧急疏散系统国际规则；

b） 商业和政府数据安全要求；

c） 公共媒体和现场演出数据安全要求。

开放式控制架构的安全取决于使用的通信协议。

例如：在使用TCP/IP通讯和GY/T 322.3—2019协议的网络中，控制数据的安全应使用TCP/IP的传输层安全（TLS）协议来提供授权和加密，见GY/T 322.3—2019。

控制数据的安全性可在全局范围内启用或禁用，即整个符合开放式控制架构的网络。符合开放式控制架构的网络中安全和不安全的设备不应混用。

注：访问控制超出开放式控制架构的范围。访问控制规定了设备、对象、对象特征和对象值范围哪些会受用户的影响。如果需要访问控制，则可在应用中实现，开放式控制架构的安全可用于实施安全。

10 并发控制

开放式控制架构应支持同时使用多个控制器的应用，其中特定对象可以接收多个控制器的指令。

在这种应用中，某些应用控制事件需要交换多个控制消息。

因此存在竞争条件的可能。为了防止竞争，符合开放式控制架构的设备应通过下文所述的简单对象锁定机制支持单线程。

开放式控制架构的锁阻止除锁定持有者外其他所有对对象的访问。

锁的接口应在OcaRoot类中定义，并应被其他每个开放式控制架构的类继承。如果需要，可按照如下描述定义不可锁定对象。

开放式控制架构锁定规则如下：

a） 每个开放式控制架构的对象应定义锁接口。

b） 对象可实现为可锁的或不可锁的。不可锁对象应对所有锁相关命令返回未实现状态（通过响应消息见 4.3）。

c） 同一时刻一个可锁对象最多被一个远程对象（锁持有者）锁定。

d） 锁应在设备复位时清除。

e） 控制器与设备的通信宜持续监控（见 11.2）。不管任何原因，当控制器和设备间通信失败时，设备应自动撤消控制器设置的所有锁。

f） 如果设备的其他对象没有被锁定，则可通过锁定设备管理单元对象来锁定整个设备。

注：开放式控制架构不包括完整的锁管理功能。例如，没有死锁检测，没有非独占锁，没有多级锁机制。如果需要，由应用实现这些功能。

11 可靠性

11.1 概述

开放式控制架构的目的是支持在足够可靠水平级别下运行的网络，满足：

a） 紧急疏散系统国际法规；

b） 商业和政府的持续运行和鲁棒性的要求；

c） 公共媒体与现场演出的持续运行和鲁棒性的要求。

“可靠性”指可用性和鲁棒性的累积效应。

11.2 可用性

11.2.1 keep-alive消息

符合开放式控制架构的网络可用性宜通过持续的设备监管来实现，该监管使用被定义成网络协议规范的周期性keep-alive消息。此消息的特性取决于正在使用的特定网络协议。

通过从应用到网络的完整路径发送和返回的本地网络协议消息，设备可自我监测。

11.2.2 有效重新初始化

当出现错误和配置变化时，协议实现宜快速和高效地重新初始化受影响的网络设备。

11.3 鲁棒性

为了实现鲁棒，开放式控制架构应包含确认操作机制，且应定义对PDU丢失和设备故障有抵抗力的容错机制。

符合开放式控制架构的协议实现可以使用特定类型网络的鲁棒机制。

示例：

当GY/T 322.3—2019定义的协议在以太网中运行时，其实现可使用生成树协议，增加对网络链路故障的抵抗力。

12 设备复位

符合开放式控制架构的设备可支持设备复位功能，以便从灾难性错误中恢复。设备复位应将受影响的设备状态快速恢复到上电时的状态。设备复位应删除所有设备初始化的数据（如，路由信息）。

注1：预计设备复位仅在极端情况下使用。

设备收到设备复位命令后应进行设备复位。设备复位命令应是包含128位复位校验码的特定消息。校验码应与设备中预置值匹配。如果不匹配，设备复位命令应无效。

复位校验码应通过特定的开放式控制架构的命令设置，并应在上电复位时清除存储的校验码。如果最近一次上电复位后，设备没收到复位校验码，则应忽略所有设备复位命令。

注2：由于成功的设备复位命令会引起上电复位，因此设备复位命令将导致受影响设备的复位校验码丢失。

设备复位命令应通过快速消息传递服务发送，并且当使用的协议许可时可是组播。

注3：当不使用开放式控制架构的安全时，由于校验码可能随时被重置，复位校验码机制是不安全的。只有当开放式控制架构的全局安全启用时，复位机制的完全安全才可用。对不安全实现，复位校验码机制用于减少偶发系统复位的概率。

13 固件和软件更新

13.1 概述

开放式控制架构应定义通过网络可靠更新设备固件和软件的机制。这种机制的实现应可选。下文中“更新器”应指驱动固件更新过程的符合开放式控制架构的控制器。

当实现开放式控制架构的网络更新时：

a) 更新应实现保证设备从升级失败中恢复的方法。因此，设备必须有一个受保护的启动下载器，即使设备程序存储区包含无效映像或无映像时，它也能继续运行。
b) 固件或软件的更新安全应通过控制数据的正常安全机制处理（见第 9 章）。
c) 符合开放式控制架构的固件更新机制的实现宜确保固件升级被正确标记和控制。这类控制的细节超出了开放式控制架构的范围。

13.2 更新类型

开放式控制架构应支持每个设备有多个固件组件，且应允许设备支持表11中4种更新类型的任何一种。

表11 更新类型

更新类型	描述
1	基于事务的完全更新：其中新接收的组件固件映像只有在所有组件映像加载成功时才应投入使用。任一组件更新失败，设备应恢复到所有旧映像
2	基于事务的部分更新：其中每个新接收的组件固件映像在其特定加载成功后应立即投入使用。如果组件映像加载不成功，设备应恢复到组件的先前映像
3	不基于事务的保护更新：传入的组件固件映像应立即覆盖当前固件。如果映像加载失败，设备应从内部只读存储器重新加载故障保护映像（有时称为黄金映像）；这些映像应提供足够功能以便手动更新和/或恢复到更新前软件版本
4	不基于事务的无保护更新：传入的更新固件映像应立即覆盖当前固件，没有故障保护机制从失败上载中恢复
注 1：基于事务的部分更新成功后可能导致设备中既有旧映像组件又有新映像组件。如果设备实现不能容忍这种情况，可能导致设备故障和不可再更新。 注 2：不基于事务的无保护更新可导致设备故障和不可再更新。	

13.3 更新模式

开放式控制架构应支持如下所示两种固件映像数据更新模式：
a） 主动更新，更新器应调用设备中特定方法上载固件映像数据至该设备；
b） 被动更新，设备应从更新器指定的源中下载固件映像数据。

13.4 更新机制

13.4.1 概述

所有固件更新应由OcaFirmwareManager对象管理，它是每个符合开放式控制架构的设备的强制性管理单元。

主动更新过程与被动更新过程不同。每种更新过程都支持13.2中描述的4种更新类型。更新类型的选择应是设备实现的一个选项。

13.4.2 主动更新

设备的主动更新应按照如下步骤进行：
a) 更新器应调用OcaFirmwareManager.StartUpdateProcess()方法。

b) 对每个要更新的固件组件映像，更新器应：

1) 调用 OcaFirmwareManager.BeginActiveImageUpdate()方法启动组件更新；

2) 根据需要多次调用 OcaFirmwareManager.AddImage()方法上载完整固件组件映像到设备；

3) 调用 OcaFirmwareManager.VerifyImage()方法校验上传固件映像的完整性；

4) 调用 OcaFirmwareManager.EndActiveImageUpdate()方法使设备完成特定组件映像的更新。

c) 更新器应调用 OcaFirmwareManager.EndUpdateProcess() 方法使设备完成更新过程。

在基于事务的完全更新情形中，只有当所有上载成功后，设备应将上载好的映像按照步骤c)运行。

在基于事务的部分更新情形中，设备应将上载好的每个校验成功的映像按照步骤b)中的4)运行。

在不基于事务的保护和无保护更新情形中，设备应将上载好的每个映像的片段根据步骤b)中的3)的返回结果运行。

13.4.3 被动更新

设备的被动更新应按照如下步骤进行：

a) 更新器应调用 OcaFirmwareManager.StartUpdateProcess()方法。

b) 对每个要更新的固件组件映像：

1) 更新器应调用 OcaFirmwareManager.BeginPassiveComponentUpdate()方法，传递网络文件服务器的主机名，以及设备从文件服务器下载的固件映像组件的文件名；

2) 设备应下载特定固件映像文件。

c) 当所有映像文件下载后，更新器应调用 OcaFirmwareManager.EndUpdateProcess()方法使设备完成更新过程。

在基于事务的完全更新情形中，只有当所有上载成功后，设备应将上载好的映像按照步骤c)运行。

在基于事务的部分更新情形，设备应在步骤b)中的2)后启用每个上载好的映像。

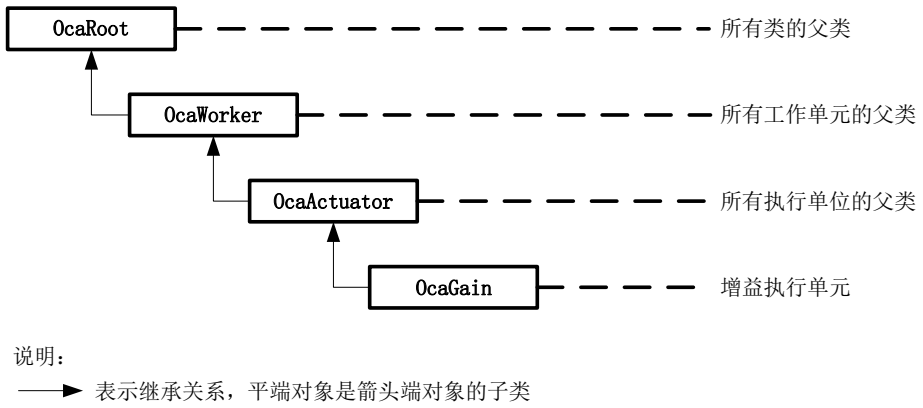
在不基于事务的保护和无保护更新情形中，设备应在步骤b)中的2)后启用每个上载好的映像。

附 录 A
（资料性附录）
执行器实例

A.1 概述

本附录用OcaGain类为例，详细说明类的构建。OcaGain是一个执行器，但其运行方式与所有类（工作单元、管理单元或代理）相同。

OcaGain继承类树如图A.1所示。注意最上面的三个类（OcaRoot、OcaWorker、OcaActuator）是抽象类，永远不会被独立地实例化。它们存在于类树中，分别表示对象、工作单元和执行器的某些共同特征。



图A.1 OcaGain 继承类树

A.2 属性、方法和事件

OcaGain的属性、方法和事件是它的特有要素和从其父类OcaActuator、OcaWorker、OcaRoot继承过来的要素的结合。

表A.1给出了OcaGain的全部属性集合。属性的访问通过调用表A.2所示的方法实现。可被OcaGain触发的事件见表A.3。

如表A.1～表A.3所示，可调用的方法集相对较大—大于简单设备的需求。开放式控制架构定义一种“未实现”状态值，当一个设备当前还没有实现该方法或该方法的某些选项时，对象可返回“未实现”状态值。

表A.1 OcaGain 属性

属性名	ID	数据类型	访问权限	描述	所在类
Object Number	01p03	ONo	RO	OcaGain 实例化对象号	OcaGain
Lockable	01p04	Boolean	RO	如果对象能被锁定，该值为 True	OcaRoot
Role	01p05	String	RO	对象在设备中的作用，例如“通道 1 增益”	OcaRoot

表 A.1（续）

属性名	ID	数据类型	访问权限	描述	所在类
Enabled	02p03	Boolean	RW	如果对象可用，该值为 True；如果对象不可用或属性无效，该值为 false	OcaWorker
Ports	02p04	Array of structures	DD	对象输入输出端口的集合	OcaWorker
Label	02p05	String	RW	对象在系统中的用途，例如“ElVis Vocal Gain”	OcaWorker
Owner	02p06	Ono	RO	所在的块对象号	OcaGain
ClassID	04p01	ClassID	RO	OcaGain 类的 ID	OcaGain
Class Version	04p02	Integer	RO	OcaGain 类的版本号	OcaGain
Gain	04p03	Float	RW	以分贝（dB）为单位的增益值	OcaGain
注：“RO”表示只读，“RW”表示可读写，“DD”表示设备相关。					

表A.2 OcaGain 的方法

方法名	ID	描述	所在类
GetClassIdentification()	01m01	返回 ClassID 和版本	OcaRoot
GetLockable()	01m02	返回 Lockable 属性值	OcaRoot
Lock()	01m03	锁定对象	OcaRoot
Unlock()	01m04	解锁对象	OcaRoot
GetRole()	01m05	返回 Role 属性值	OcaRoot
GetEnable()	02m01	返回 Enabled 属性值	OcaWorker
SetEnabled(…)	02m02	设置 Enabled 属性值	OcaWorker
AddPort(…)	02m03	为对象增加一个信号端口	OcaWorker
DeletePort()	02m04	从对象删除一个信号端口	OcaWorker
GetPorts()	02m05	返回对象的信号端口列表	OcaWorker
GetPortName()	02m06	返回一个信号端口的名称	OcaWorker
SetPortName()	02m07	设置一个信号端口的名称	OcaWorker
GetLabel()	02m08	返回 Lable 属性值	OcaWorker
SetLable(…)	02m09	设置 Lable 属性值	OcaWorker
GetOwner()	02m10	返回包含块的 Ono	OcaWorker
GetGain()	04m01	返回 Gain 属性值	OcaGain
SetGain()	04m02	设置 Gain 属性值	OcaGain

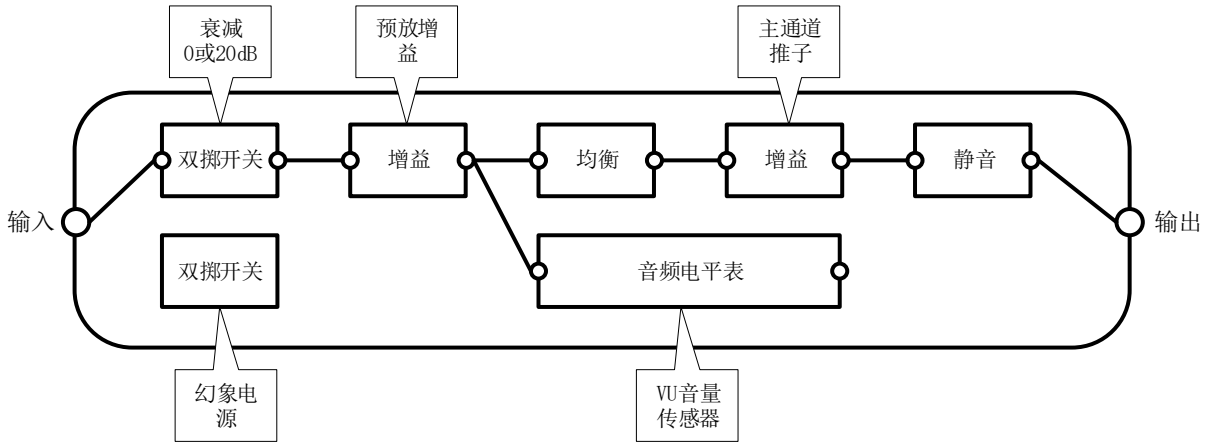
表A.3 OcaGain 的事件

事件名	ID	描述	所在类
PropertyChanged(…)	01e01	当对象的属性值改变时触发	OcaRoot

附 录 B
（资料性附录）
块实例

B.1 简单传声器通道

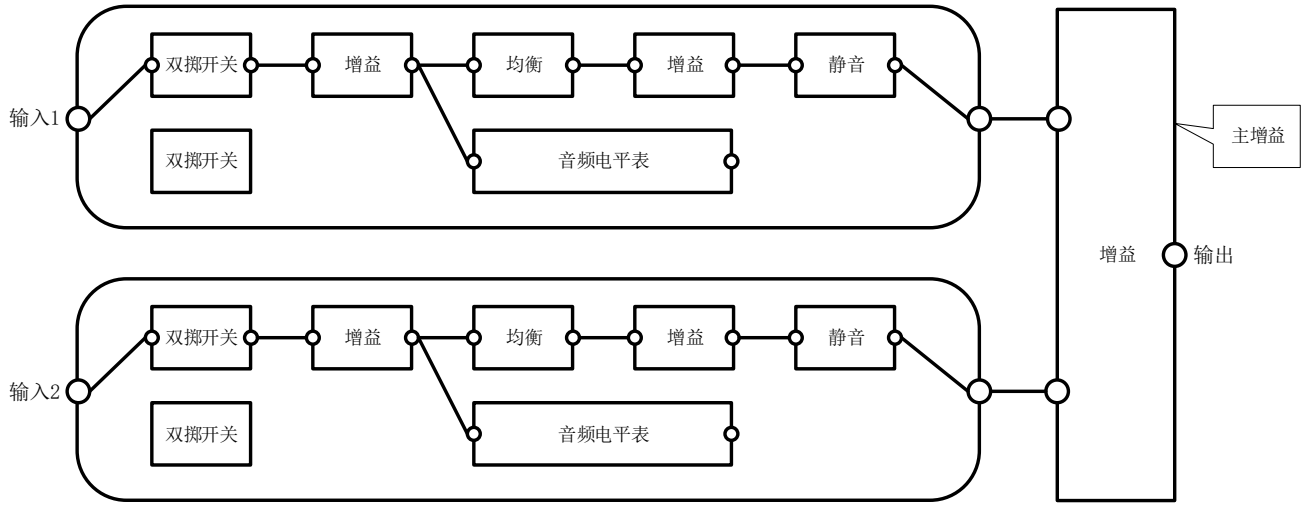
图B.1表示了典型调音台的传声器通道信号流图。



图B.1 简单传声器通道信号流图

B.2 两通道传声器调音台

图B.2说明在较大组件中块的使用。两个B.1的传声器通道块和另一个增益控制组合构成一个两通道传声器调音台。



图B.2　二通道传声器调音台

这种情况下，主增益对象有两个输入端口用于混音功能。

这里没有明确的对象用于混音总线和求和放大器。这个简单调音台，混音总线和求和放大器没有任何被远程控制的参数。

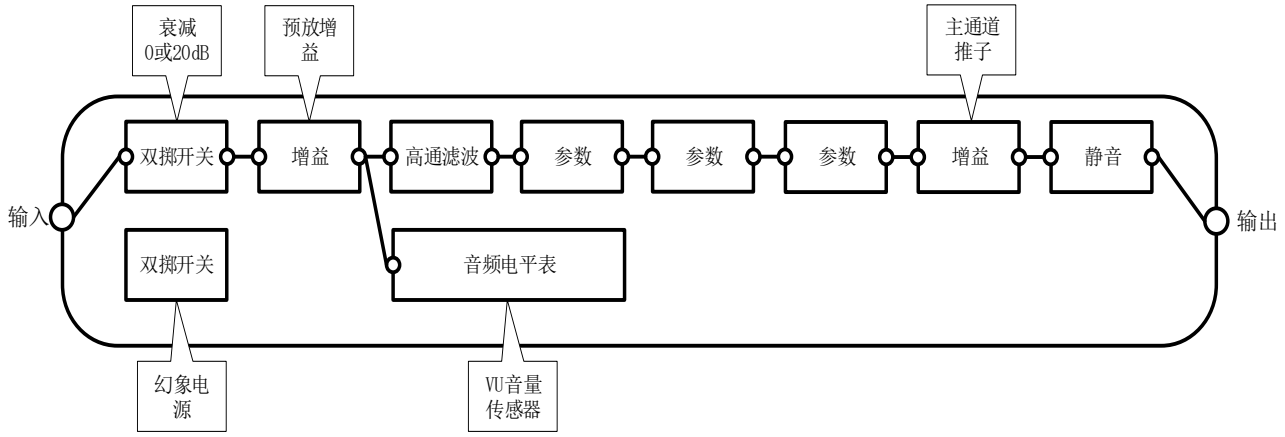
因此不需要一个开放式控制架构的相应对象。更复杂的调音台可能有与求和子系统相关的监控功能。这种情况下，信号流需包含一个明确的求和点。

B.3　采用嵌套块的调音台

考虑图B.1的传声器通道模型中包含一个均衡器对象。典型的传声器通道均衡器包含一个高通滤波器，以及后续三个参数均衡器。

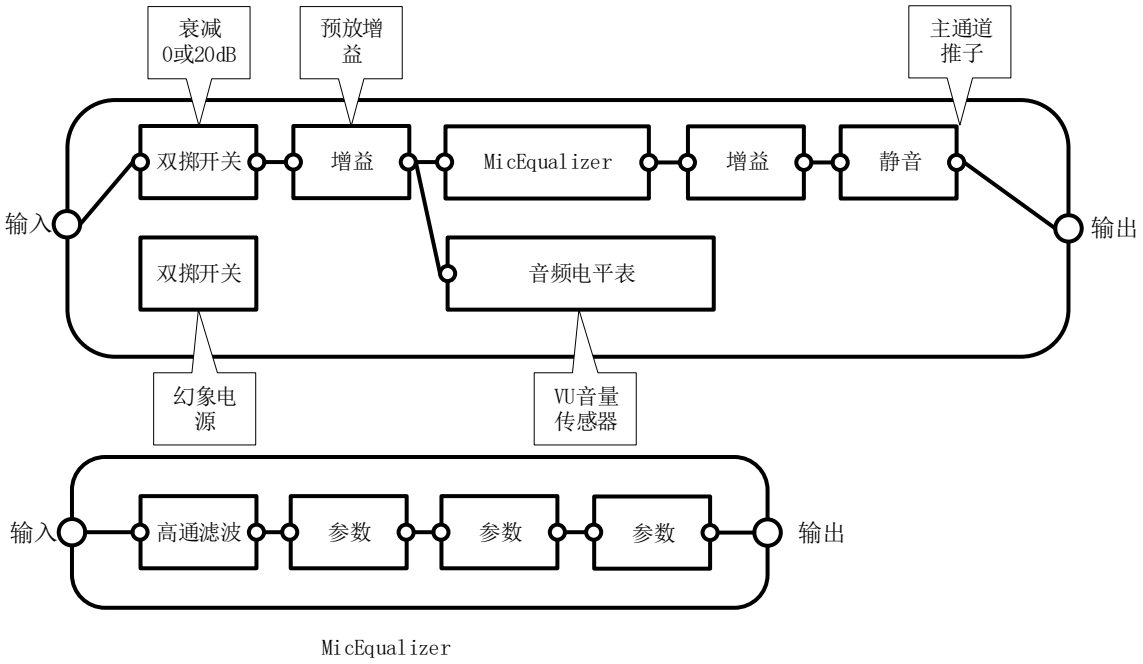
开放式控制架构定义了高通滤波器或参数均衡器的对象类。可以用1个OcaFilterClassical实例，以及后续3个OcaFilterParametric实例，来实现典型传声器通道均衡器建模。

它们可以依次简单地增加到传声器通道中，如图B.3所示。



图B.3　包含均衡处理的传声器通道

也可以定义一个名为MicEqualizer的块，此块包含所有均衡器对象，并将其嵌套到传声器通道块中，如图B.4所示。



图B.4　包含 MicEqualizer 块的传声器通道

这种嵌套的方式更容易用在可重配置设备中。它可使控制器实现更加简单，特别是同样的均衡器在设备的不同部分，或在其他产品中被应用。

附 录 C
（资料性附录）
网络连接管理示例

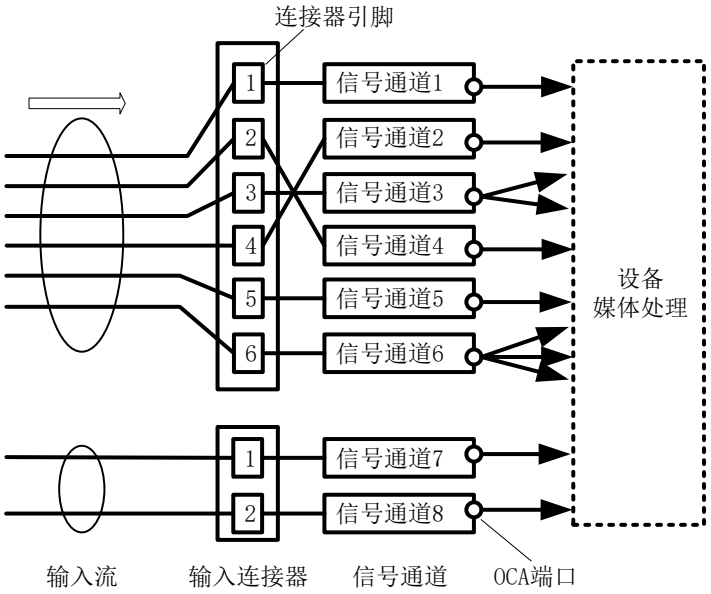
C.1 基于流连接示例

C.1.1 概述

开放式控制架构允许多个基于流的连接场景，在下文举例说明。下文这些例子不基于任何特定的传输网络体系结构；它们用来说明开放式控制架构能够管理的基于流连接的范围的抽象案例。

C.1.2 场景1：输入

在这个场景中，两个输入流连接到两个连接器，共形成8个信号通道，如图C.1所示。开放式控制架构不限制设备可具有的输入连接器数量和大小。

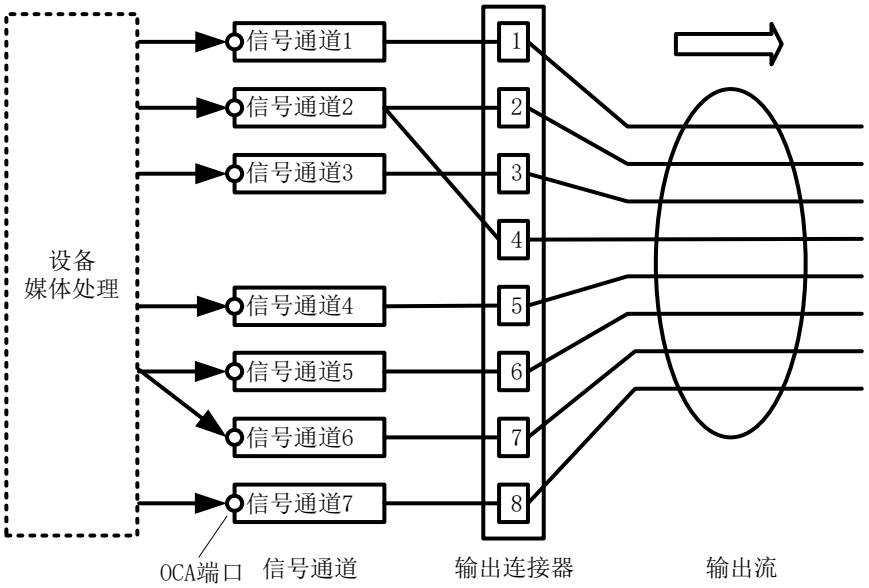


图C.1 基于流的输入示例

图C.1中连接器的设备侧对输入进行了重映射，连接器原先的引脚2和4分别映射至信号通道的4和2。根据开放式控制架构的信号流机制，设备侧的信号通道可以再次重映射。

C.1.3 场景2：简单输出

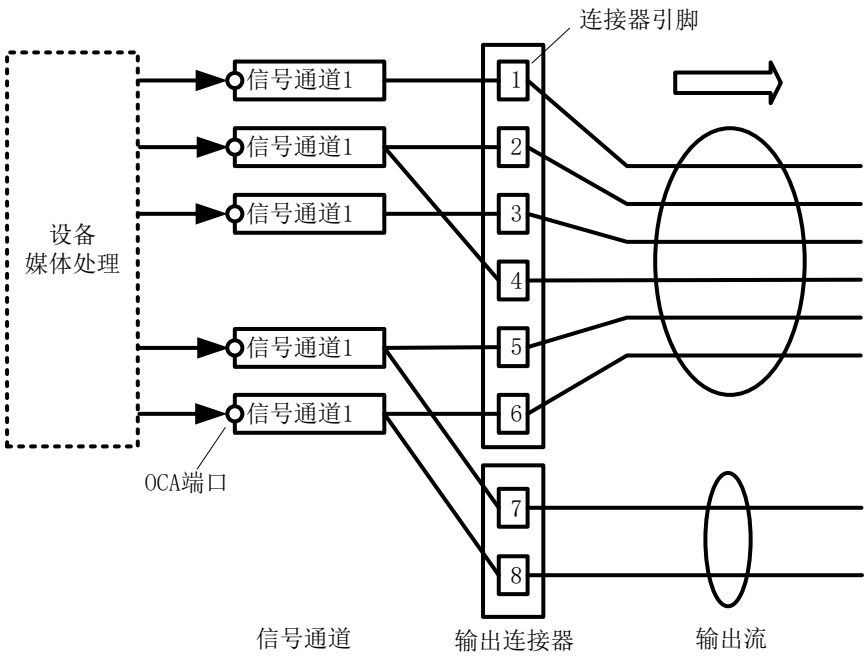
在这个场景中，7个信号通道生成了8通道的输出流，如图C.2所示。信号通道2驱动两个连接器引脚。根据开放式控制架构的信号流机制，设备侧的信号通道可以重映射。



图C.2　基于流的简单输出示例

C.1.4　场景3：多输出连接器

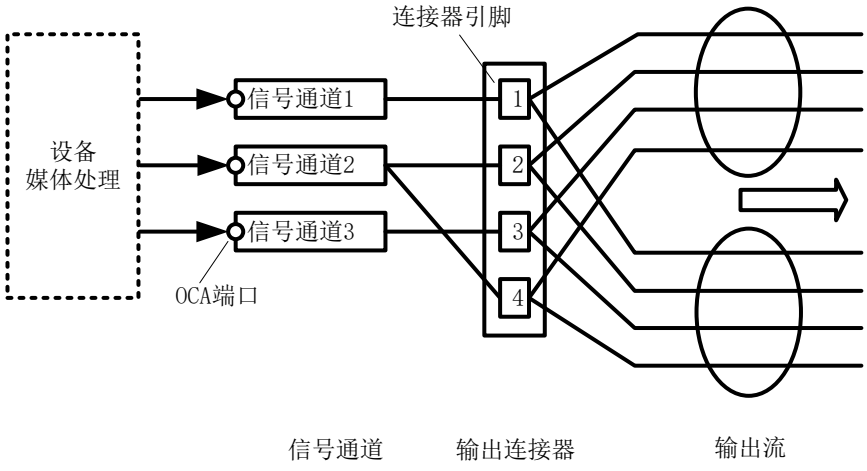
在这个场景中，输出信号通道中的两个输出信号通道驱动多个输出连接器，生成了两个输出流，如图C.3所示。



图C.3　基于多流的输出连接器示例

C.1.5　场景4：来自于一个连接器的多个输出流

在这个场景中，2个相同的输出流由同一个输出连接器驱动，如图C.4所示。

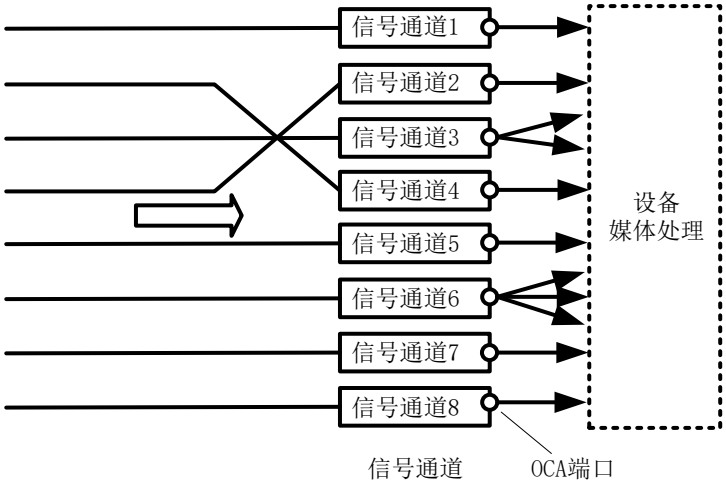


图C.4　来自于一个连接器的多个输出流

C.2　基于通道连接示例

C.2.1 场景5:输入

在这个场景中，8个输入通道和8个信号通道连接。根据开放式控制架构的信号流机制，设备侧的信号通道可以重映射，如图C.5所示。

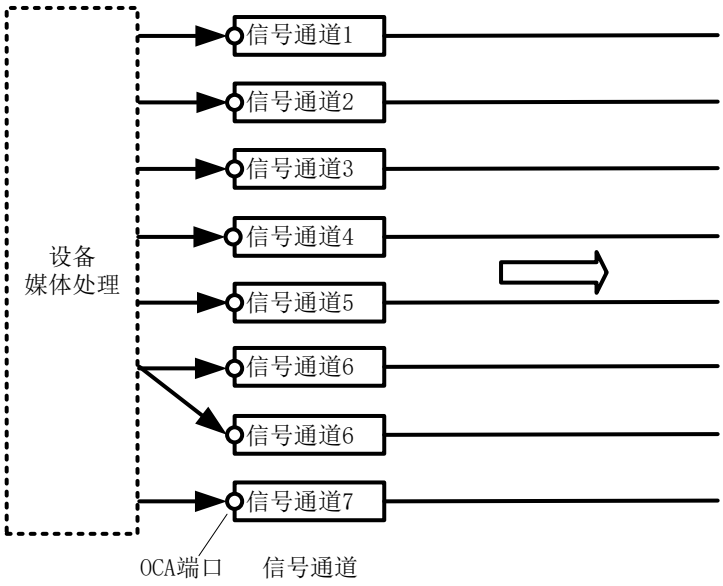


图C.5 基于通道的输入

与基于流的模式不同，基于通道模式中的信号通道不与连接器的引脚连接。实际上，它们直接与网络信号通道连接。OcaNetworkSignalChannel类包含用于识别和描述连接的网络信号通道的属性。

C.2.2 场景6：输出

在这个场景中，8个信号通道生成8个输出通道。按照开放式控制架构的信号流机制，设备侧的信号通道可以重映射，如图C.6所示。


设备
媒体处理
信号通道1
信号通道2
信号通道3
信号通道4
信号通道5
信号通道6
信号通道6
信号通道7
OCA端口
信号通道

图C.6　基于通道的输出

中 华 人 民 共 和 国
广播电视行业标准
网络音频应用的开放式控制架构
第 1 部分：框架
GY/T 322.1－2019
*
国家广播电视总局广播电视规划院出版发行
责任编辑：王佳梅
查询网址：www.abp2003.cn
北京复兴门外大街二号
联系电话：（010）86093424　　86092923
邮政编码：100866