# MIDI-CI Property Exchange Foundational Resources

DeviceInfo, ChannelList, JSONSchema

**MIDI Association Document: M2-105-UM**

Document Version 1.1.1
Draft Date May 31, 2023

**Published June 15, 2023**

Developed and Published By
**The MIDI Association**
and
**Association of Musical Electronics Industry (AMEI)**

## PREFACE

MIDI-CI Property Exchange Foundational Resources

Property Exchange is part of the MIDI-CI specifications first released in 2018. Property Exchange is a method for sending JSON over SysEx between two Devices to get and set Device properties. Each MIDI Device is unique and provides an experience different from another Device. Property Exchange allows you to discover and use almost any Device in a consistent way.

This document describes the Property Data for these Resources. For information on how to transmit and receive Property Data over SysEx please see the MIDI-CI ***[MA02]*** and Common Rules for MIDI-CI Property Exchange ***[MA04]***.

http://www.amei.or.jp

https://www.midi.org

# Version History

**Table 1 Version History**

| Publication Date | Version | Changes |
|---|---|---|
| November 24, 2020 | 1.0.1 | Initial release |
| June 15, 2023 | 1.1 | Updated ChannelList Resource to support Function Blocks and improve Clusters |

# Contents

# Figures

# Tables

# 1 Introduction

## 1.1 Executive Summary

Property Exchange is part of MIDI 2.0 and the MIDI-CI specifications for autoconfiguration, reducing manual configuration and enabling easier setup for users.

Foundational Resources are the basis of Property Exchange. The properties provided by these Resources are common to most MIDI Devices and most valuable as core information for autoconfiguration. These properties include fundamental Device identifiers and information about the MIDI Channels which are actively in use by the Device.

Property Exchange Resources are defined sets of JSON data inside MIDI System Exclusive, used to get and set Device properties.

It is strongly recommended that all MIDI 2.0 Devices support these Property Exchange Resources. Other Property Exchange mechanisms and Resources have strong dependencies on the Resources defined in this specification.

## 1.2 Background

Property Exchange is part of the MIDI Capability Inquiry (MIDI-CI) ***[MA02]*** specification and MIDI 2.0. Property Exchange is a method for getting and setting various data, called Resources, between two Devices. Resources are exchanged inside two payload fields of System Exclusive Messages defined by MIDI-CI, the Header Data field and Property Data field. This document defines only the contents of the Header Data and Property Data fields. For information on how to transmit and receive these Resource payloads inside MIDI-CI System Exclusive messages, see the MIDI Capability Inquiry specification ***[MA02]*** and Common Rules for MIDI-CI Property Exchange specification ***[MA04]***.

This specification defines three Foundational Resources: **DeviceInfo**, **ChannelList**, and **JSONSchema**

## 1.3 References

### 1.3.1 Normative References

| | |
|---|---|
| [COMM01] | ***CommonMark Spec***, Version 0.28, https://spec.commonmark.org/0.28/ |
| [ECMA01] | ***The JSON Data Interchange Syntax***, ECMA-404, https://www.ecma-international.org/publications/standards/Ecma-404.htm |
| [MA01] | ***Complete MIDI 1.0 Detailed Specification***, Document Version 96.1, Third Edition, Association of Musical Electronics Industry, http://www.amei.or.jp/, and The MIDI Association, https://www.midi.org/ |
| [MA02] | ***M2-101-UM MIDI Capability Inquiry (MIDI-CI)***, Version 1.2, Association of Musical Electronics Industry, http://www.amei.or.jp/, and The MIDI Association, https://www.midi.org/ |
| [MA03] | ***M1-100-UM MIDI Polyphonic Expression,*** Version 1.1, Association of Musical Electronics Industry, http://www.amei.or.jp/, and The MIDI Association, https://www.midi.org/ |
| [MA04] | ***M2-103-UM Common Rules for Property Exchange***, Version 1.1, Association of Musical Electronics Industry, http://www.amei.or.jp/, and The MIDI Association, https://www.midi.org/ |
| [MA05] | ***M2-104-UM Universal MIDI Packet (UMP) Format and MIDI 2.0 Protocol***, Version 1.1, Association of Musical Electronics Industry, http://www.amei.or.jp/, and The MIDI Association, https://www.midi.org/ |
| [MA06] | ***M2-107-UM MIDI-CI Property Exchange ProgramList Resource***, Version 1.0, Association of Musical Electronics Industry, http://www.amei.or.jp/, and The MIDI Association, https://www.midi.org/ |
| [MA07] | ***M2-108-UM MIDI-CI Property Exchange Channel Resources***, Version 1.0, Association of Musical Electronics Industry, http://www.amei.or.jp/, and The MIDI Association, https://www.midi.org/ |
| [MA08] | ***M2-102-U Common Rules for MIDI-CI Profiles***, Version 1.1, Association of Musical Electronics Industry, http://www.amei.or.jp/, and The MIDI Association, https://www.midi.org/ |

### 1.3.2 Informative References

No informative references.

## 1.4 Terminology

### 1.4.1 Definitions

**AMEI:** Association of Musical Electronics Industry. Authority for MIDI Specifications in Japan.

**Cluster:** A collection of related MIDI Channels whether because the Device is operating in a multi-channel mode (such as Mode 4 Omni Off/Mono) or for any other reason the Channels have some related functionality.

**Device:** An entity, whether hardware or software, which can send and/or receive MIDI messages.

**Foundational Resource:** A Resource which provides core Properties of a Device which are critical or highly valuable to properly implement numerous other Resources.

**Group:** A field in the UMP Format addressing some UMP Format MIDI messages (and some UMPs comprising any given MIDI message) to one of 16 Groups. See the M2-104-UM Universal MIDI Packet (UMP) Format and MIDI 2.0 Protocol specification ***[MA05]***.

**Initiator:** One of two MIDI-CI Devices with a bidirectional communication between them. Initiator has the management role of setting and negotiating parameters for interoperability between the two Devices. The primary goal of Initiator is usually (but not strictly required to be) configuring two Devices for subsequent communication from Initiator as MIDI transmitter to Responder as MIDI receiver. The role of Initiator and Responder may alternate between the two MIDI-CI Devices. Either MIDI-CI Device may initiate a MIDI Transaction (act as Initiator) at any time. Also see Responder.

**Inquiry:** A message sent by an Initiator to begin a Transaction.

**JSON:** JavaScript Object Notation as defined in ***[ECMA01]***.

**List Resource:** A specific type of Resource that provides a list of objects in a JSON array.

**MA:** MIDI Association.

**MIDI 1.0 Protocol:** Version 1.0 of the MIDI Protocol as originally specified in ***[MA01]*** and extended by MA and AMEI with numerous additional MIDI message definitions and Recommended Practices. The native format for the MIDI 1.0 Protocol is a byte stream, but it has been adapted for many different transports. MIDI 1.0 messages can be carried in UMP packets. The UMP format for the MIDI 1.0 Protocol is defined in the M2-104-UM Universal MIDI Packet (UMP) Format and MIDI 2.0 Protocol specification ***[MA05]***.

**MIDI 1.0 Specification:** Complete MIDI 1.0 Detailed Specification, Document Version 96.1, Third Edition ***[MA01]***.

**MIDI 2.0:** The MIDI environment that encompasses all of MIDI 1.0, MIDI-CI, Universal MIDI Packet (UMP), MIDI 2.0 Protocol, MIDI 2.0 messages, and other extensions to MIDI as described in AMEI and MA specifications.

**MIDI 2.0 Protocol:** Version 2.0 of the MIDI Protocol. The native format for MIDI 2.0 Protocol messages is UMP as defined in M2-104-UM Universal MIDI Packet (UMP) Format and MIDI 2.0 Protocol specification ***[MA05]***.

**MIDI-CI:** MIDI Capability Inquiry, a specification published by The MIDI Association and AMEI.

**MIDI-CI Device:** A Device that has the ability to act as a Responder that replies to inquiries received from an Initiator. The ability to act as an Initiator is recommended but optional.

**MIDI-CI Transaction:** A Transaction using a set of MIDI-CI messages that includes an Inquiry sent by an Initiator and a reply to the Inquiry returned by the Responder. The Responder's reply to an Inquiry might be a single message that satisfies the Inquiry, a set of multiple messages that satisfy the Inquiry, or an error message. See also Transaction.

**MIDI Association:** Authority for MIDI specifications worldwide except Japan. See also MMA.

**MIDI Manufacturers Association:** a California nonprofit 501(c)6 trade organization, and the legal entity name of the MIDI Association.

**MMA:** MIDI Manufacturers Association.

**MPE Zone:** A group of contiguous MIDI Channels comprising a Master Channel and one or more Member Channels that are used with MIDI Polyphonic Expression. See MIDI Polyphonic Expression ***[MA03]***.

**PE:** Property Exchange.

**Profile:** An MA/AMEI specification that includes a set of MIDI messages and defined responses to those messages. A Profile is controlled by MIDI-CI Profile Negotiation Transactions. A Profile may have a defined minimum set of mandatory messages and features, along with some optional or recommended messages and features. See the MIDI-CI specification ***[MA02]*** and the Common Rules for MIDI-CI Profiles ***[MA08]***.

**Program:** A set of Device parameters which is selectable by Bank Select and Program Change messages.

**Program Collection:** A grouping of Programs with some common trait (bank, category, instrument, synthesis engine, presets, etc).

**Property:** A JSON key-value pair used by Property Exchange, for example `"channel": 1`.

**Property Exchange:** A set of MIDI-CI Transactions by which one Device may access Property Data from another Device.

**Property Key:** The key in a JSON key-value pair used by Property Exchange.

**Property Value:** The value in a JSON key-value pair used by Property Exchange.

**Protocol:** There are two defined MIDI Protocols: the MIDI 1.0 Protocol and the MIDI 2.0 Protocol, each with a data structure that defines the semantics for MIDI messages. See ***[MA01]*** and ***[MA05]***.

**Resource:** A defined collection of one or more PE Properties with an associated inquiry to access its Properties.

**Responder:** One of two MIDI-CI Devices with a bidirectional communication between them. The Responder is the Device that receives an Inquiry message from an Initiator Device as part of a MIDI-CI Transaction and acts based on negotiation messages managed by the Initiator Device. Also see Initiator.

**Transaction:** An exchange of MIDI messages between two MIDI Devices with a bidirectional connection. All the MIDI messages in a single Transaction are associated and work together to accomplish one function. The simplest Transaction generally consists of an inquiry sent by one MIDI Device and an associated reply returned by a second MIDI Device. A Transaction may also consist of an inquiry from one MIDI Device and several associated replies from a second MIDI Device. A Transaction may be a more complex set of message exchanges, started by an initial inquiry from one MIDI Device and multiple, associated replies exchanged between the first MIDI Device and a second MIDI Device. Also see MIDI-CI Transaction.

**UMP:** Universal MIDI Packet, see ***[MA05]***.

**UMP Format:** Data format for fields and messages in the Universal MIDI Packet, see ***[MA05]***.

**Universal MIDI Packet (UMP):** The Universal MIDI Packet is a data container which defines the data format for all MIDI 1.0 Protocol messages and all MIDI 2.0 Protocol messages. UMP is intended to be universally applicable, i.e., technically suitable for use in any transport where MA/AMEI elects to officially support UMP. For detailed definition see M2-104-UM Universal MIDI Packet (UMP) Format and MIDI 2.0 Protocol specification ***[MA05]***.

### 1.4.2 Reserved Words and Specification Conformance

In this document, the following words are used solely to distinguish what is required to conform to this specification, what is recommended but not required for conformance, and what is permitted but not required for conformance:

**Table 2 Words Relating to Specification Conformance**

| Word | Reserved For | Relation to Specification Conformance |
|---|---|---|
| **shall** | Statements of requirement | **Mandatory**<br>A conformant implementation conforms to all 'shall' statements. |
| **should** | Statements of recommendation | **Recommended but not mandatory**<br>An implementation that does not conform to some or all 'should' statements is still conformant, providing all 'shall' statements are conformed to. |
| **may** | Statements of permission | **Optional**<br>An implementation that does not conform to some or all 'may' statements is still conformant, providing that all 'shall' statements are conformed to. |

By contrast, in this document, the following words are never used for specification conformance statements; they are used solely for descriptive and explanatory purposes:

**Table 3 Words Not Relating to Specification Conformance**

| Word | Reserved For | Relation to Specification Conformance |
|---|---|---|
| **must** | Statements of unavoidability | Describes an action to be taken that, while not required (or at least not directly required) by this specification, is unavoidable.<br>Not used for statements of conformance requirement (see 'shall' above). |
| **will** | Statements of fact | Describes a condition that as a question of fact is necessarily going to be true, or an action that as a question of fact is necessarily going to occur, but not as a requirement (or at least not as a direct requirement) of this specification.<br>Not used for statements of conformance requirements (see 'shall' above). |
| **can** | Statements of capability | Describes a condition or action that a system element is capable of possessing or taking.<br>Not used for statements of conformance permission (see 'may' above). |
| **might** | Statements of possibility | Describes a condition or action that a system element is capable of electing to possess or take.<br>Not used for statements of conformance permission (see 'may' above). |

# 2 Three Foundational Resources

Foundational Resources provide core Properties of PE Devices and serve to enable other Resources. Many other Resources depend on or make use of Properties discovered in Foundational Resources.

Foundational Resources are the Resources which are most often used immediately following a **ResourceList** inquiry (see Common Rules for MIDI-CI Property Exchange).

This specification defines a Foundational Resource that shall be implemented by all Property Exchange Devices:

- **DeviceInfo**

This specification defines a Foundational Resource that should be implemented by all Property Exchange Devices:

- **ChannelList**

This specification also defines a Foundational Resource that shall be implemented by all Property Exchange Devices which implement Manufacturer Specific Resources which use the **`"$ref"`** Property in ResourceList schema with a Property Value in the form of `"midi+jsonschema://<JSON Schema Id>"`:

- **JSONSchema**

# 3 Foundational Resource: DeviceInfo

The **DeviceInfo** Resource provides core details about the identity of a Property Exchange Device. It contains the same data as the Device Inquiry Universal SysEx Message and the Device Identity Notification UMP message. **DeviceInfo** also includes human-readable Properties for Manufacturer, Family, Model, Version information, and more.

All Property Exchange Devices shall implement this Resource.

## 3.1 Initiator Requests Data from a Responder Using an Inquiry: Get Property Data

An Initiator may request the **DeviceInfo** Resource from a Responder using an **Inquiry: Get Property Data** message.

**Table 4 Initiator Sends Inquiry: Get Property Data**

| | |
|---|---|
| `Header Data` | `{"resource":"DeviceInfo"}` |
| `Property Data` | *none* |

## 3.2 Responder Sends a Reply to Get Property Data Message

A Responder that supports **DeviceInfo** Resource shall return an object in the Property Data using a Reply to Get Property Data Message.

**Table 5 DeviceInfo Message Data Object**

| Property Key | Property Value Type | Description |
|---|---|---|
| **`manufacturerId`** | array of numbers (integers), required | This is an array of the 3 data byte values (as integers) in the same structure as defined in MIDI-CI Discovery Messages |
| **`familyId`** | array of numbers (integers), required | This is an array of the 2 data byte values (as integers) in the same structure (LSB First) as defined in MIDI-CI Discovery Messages. If the Responder does not have a familyId, it uses [0,0]. |
| **`modelId`** | array of numbers (integers), required | This is an array of the 2 data byte values (as integers) in the same structure (LSB First) as defined in MIDI-CI Discovery Messages |
| **`versionId`** | array of numbers (integers), required | The Device version. This is an array of the 4 data byte values (as integers) in the same structure as defined in MIDI-CI Discovery Messages |
| **`manufacturer`** | string, required | The name of the Manufacturer |
| **`family`** | string, required | The product family name |
| **`model`** | string, required | The model name |
| **`version`** | string, required | The Device version. A manufacturer defined string. |
| **`serialNumber`** | string | The Device may return a unique serial number. |

The **`"familyId", "modelId"`**, and **`"versionId"`** Properties Values shall match the respective fields in the MIDI-CI Discovery Message.

**Table 6 Responder Sends Reply to Get Property Data**

| | |
|---|---|
| Header Data | `{"status":200}` |
| Property Data | `{ "version": "1.0", "manufacturerId": [125,0,0], "manufacturer": "Educational Use", "familyId": [0,0], "family": "Test Range", "modelId": [48,0], "model": "MIDI-CI Test Workbench", "versionId": [0,0,1,0], "links": [ {"resource": "X-SystemSettings"}, {"resource": "X-LocalOn"} ] }` |

## 3.3 Using "links" Property to Work with Other Resources

These Links are for Resources that affect the overall settings of the Device. Examples of this include system settings of the Device.

It is recommended that Links contained in the **DeviceInfo** Resource Property Data are presented whenever possible on the Initiator, in order to present the user with additional options for accessing further properties of the Device.

See Common Rules for MIDI-CI Property Exchange ***[MA04]***.

## 3.4 ResourceList Integration for DeviceInfo

**Table 7 Minimal entry in ResourceList**

| | |
|---|---|
| Property Data | `[ {"resource": "DeviceInfo"} ]` |

**Table 8 Full version with default settings**

| | |
|---|---|
| Property Data | `[ { "resource": "DeviceInfo", "canGet": true, "canSet": "none", "canSubscribe": false, "schema": { "type": "object", "title": "Device Information", "$ref": "http://schema.midi.org/property-exchange/M2-105-S_v1-0_DeviceInfo.json" } } ]` |

# 4 Foundational Resource: ChannelList

**ChannelList** is a List Resource which describes the current MIDI Channels in use across the whole Device or, in the case of a Device using the Universal MIDI Packet Format, the current MIDI Channels in use across one Function Block of the Universal MIDI Packet Format. It describes the current Channel or Cluster of Channels, Program, and other properties.

It is strongly recommended that all Property Exchange Devices implement this Resource.

## 4.1 Subscribable Resource

If a user can update this Resource Property Data on the Device, then this Resource should be subscribable. The mechanism for subscribing to Resources is defined in the Common Rules for Property Exchange ***[MA04]***.

## 4.2 Initiator Requests Data from a Responder Using an Inquiry: Get Property Data

An Initiator may request the **ChannelList** Resource from a Responder using an **Inquiry: Get Property Data** message.

**Table 9 Initiator Sends Inquiry: Get Property Data**

| | |
|---|---|
| `Header Data` | `{"resource":"ChannelList"}` |
| `Property Data` | *none* |

## 4.3 Responder Sends a Reply to Get Property Data Message

A Responder that supports **ChannelList** Resource shall return an object in the Property Data using a Reply to Get Property Data Message.

**Table 10 ChannelList Message Data Object**

| Property Key | Property Value Type | Description |
|---|---|---|
| **`title`** | string, required | The name of the Channel being described |
| **`channel`** | number (integer) (1-256, required) | This is the Channel being described by this entry in the list of Channels. Channel numbering starts on the first channel on the first Group in the Function Block and continues across Groups in the Function Block. For example, if a Function Block contains 3 Groups it has Channels 1-48<br>An optional Cluster may exist, associated with the **`"channel"`**. |
| **`programTitle`** | string | The name of the Program currently active on this Channel |
| **`bankPC`** | array of number (integers) | This is a 3 item array containing the Bank MSB, Bank LSB and Program Change for the current Program |
| **`clusterChannelStart`** | number (integer) (1-256) | This is the starting Channel of the Cluster. Cluster is always described by the lowest Channel. |
| **`clusterLength`** | number (integer) (1-256) | This is the length of the Cluster including the **`"clusterChannelStart"`**. |
| **`clusterMidiMode`** | number (integer) (1-4) | This is the MIDI Mode of the Cluster |

| | | |
|---|---|---|
| | default: 3 (Omni Off, Poly) | |
| **`clusterType`** | enum<br>"other","profile","mpe1"<br>default: "other" | Declare the type of Cluster. This aids in identifying Basic, Global and Manager Channels.<br>*Note: `"mpe1"` does not include the MIDI-CI Profile version of MPE. In the case of an MPE Profile, set to `"profile"`.* |

The value of **`"clusterChannelStart"`** may be the same as **`"channel"`**, or may be different from **`"channel"`**, depending on Device application (See ***Section 4.4***, Manager, Global, and Basic Channels and ***Section 4.6***, Examples).

**Table 11 Responder Sends Reply to Get Property Data**

| | |
|---|---|
| `Header Data` | `{"status":200}` |
| `Property Data` | <pre>[<br>    {<br>        "title": "Lead",<br>        "channel": 1,<br>        "programTitle": "Piano + Strings",<br>        "bankPC": [1,0,76]<br>    },<br>    {<br>        "title": "Drums",<br>        "channel": 10,<br>        "programTitle": "GM2 Jazz Drum Set",<br>        "bankPC": [0,0,33]<br>    }<br>]</pre> |

## 4.4 Manager, Global, and Basic Channels

The Channel number declared in the **`"channel"`** Property may be a Basic Channel, a Global Channel, a Manager Channel, or none of these. The rules for these Channel types and their relationships to the Channel number declared in the **`"channel"`** Property depend on whether this is a single channel (no Cluster), or if there are multiple channels (has Cluster).

*Note: Basic Channel and Global Channel are concepts in the Complete MIDI 1.0 Detailed Specification* ***[MA01]***. *A Manager Channel is a MIDI 2.0 concept which is very similar to the Global Channel defined in MIDI 1.0, with only slightly different rules and specific applications in MIDI-CI.*

1. Single Channel - No Associated Cluster: The **`"channel"`** may be a Basic Channel, or it may be both a Basic Channel and a Manager Channel.
2. Multiple Channel - Channel has Associated Cluster:
   A. If "channel" is a Basic Channel for Mode 3 (Omni Off, Poly) & Mode 4 (Omni Off, Mono), then the Basic Channel shall be the first Channel in the Cluster.
   B. If "channel" is a Global Channel or a Manager Channel, then it shall not be included as a member of its associated Cluster.

Additional rules for Devices operating in Mode 1 (Omni On, Poly) and Mode 2 (Omni On, Mono):

- The Device shall declare only a single ChannelList entry for a single Group with a Cluster that spans Channels 1-16.
- The ChannelList entry shall be on the Basic Channel.

Examples showing Manager, Global, and Basic Channels are in Section *4.6*. Also see the M2-108-UM MIDI-CI Property Exchange Channel Resources: ChannelMode, BasicChannelRx, BasicChannelTx *[MA07]* specification which defines Resources for exchanging information about Basic Channels.

## 4.5 Using "links" Property to Work with Other Resources

The **ChannelList** provides opportunities to frequently use the **"links"** Property. The links Property should refer to the Resource related to the **ProgramList** available for each entry. Other frequently linked Resources may include **CtrlList**, Manufacturer defined Resources, and others.

See Common Rules for MIDI-CI Property Exchange *[MA04]*.

### 4.5.1 Example of "links" in ChannelList

This example highlights that a Channel may have many potential Resource operations that can be performed using a mixture of MMA/AMEI and Manufacturer defined options.

**Table 12 Responder Sends Reply to Get Property Data containing Links**

| | |
|---|---|
| Header Data | {"status":200} |
| Property Data | <pre>[<br>    {<br>        "title": "Lead",<br>        "channel": 1,<br>        "programTitle": "Piano + Strings",<br>        "bankPC": [1,0,76],<br>        "links": [<br>            {<br>                "resource": "ProgramList", "resId": "general",<br>                "title": "General Programs"<br>            },<br>            {"resource": "X-MidiEffects", "resId": "singch1"}<br>        ]<br>    }<br>]</pre> |

### 4.5.2 Using "links" to ProgramList in ChannelList

Each Channel entry in a **ChannelList** declares a list of Program Collections which are available on that Channel via the "links" Property. The **ProgramList** Resource retrieves the list of available Program Collections from each **ChannelList** entry (see Section 8).

**Table 13 Responder Sends Reply to Get Property Data containing ProgramList Resource Links**

| | |
|---|---|
| Header Data | {"status":200} |
| Property Data | <pre>[<br>    {<br>        "title": "Lead",<br>        "channel": 1,<br>        "programTitle": "Piano + Strings",<br>        "bankPC": [1,0,76],<br>        "links": [<br>            {<br>                "resource": "ProgramList", "resId": "presets",<br>                "title": "Factory Presets"<br>            },<br>            {</pre> |

| | |
|---|---|
| | `"resource": "ProgramList", "resId": "user1",`<br>`"title": "User Programs"`<br>`},`<br>`{`<br>`"resource": "ProgramList", "resId": "gm2melodic",`<br>`"title": "GM2 Programs"`<br>`}`<br>`]`<br>`},`<br>`{`<br>`"title": "Drums",`<br>`"channel": 10,`<br>`"programTitle": "GM2 Jazz Drum Set",`<br>`"bankPC": [0,0,33],`<br>`"links": [`<br>`{`<br>`"resource": "ProgramList", "resId": "gm2drums",`<br>`"title": "GM2 Drum Sets"`<br>`}`<br>`]`<br>`},`<br>`]` |

In the example above, the Initiator discovers three available Program Collections on Channel 1. The Initiator may expose the available Program Collections as options for the user to select a specific Program Collection for access via **ProgramList**.

## 4.6 Further Examples of ChannelList Property Data

This section presents examples for specific conditions.

### 4.6.1 Single Channel ChannelList Entry

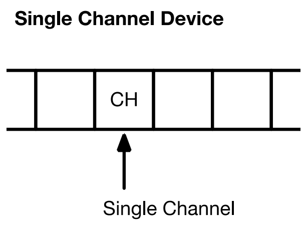

**Figure 1 Single Channel Example**

**Table 14 Property Data for a Single Channel ChannelList Entry**

| | |
|---|---|
| Header Data | `{"status":200}` |
| Property Data | `[`<br>`{`<br>`"title": "Single Channel",`<br>`"channel": 4,`<br>`"programTitle": "Church Organ 1",`<br>`"bankPC": [2,0,1],`<br>`"links": [`<br>`{"resource": "ProgramList", "resId": "B3"}`<br>`]`<br>`}`<br>`]` |

### 4.6.2 Mode 4 Device ChannelList Entry

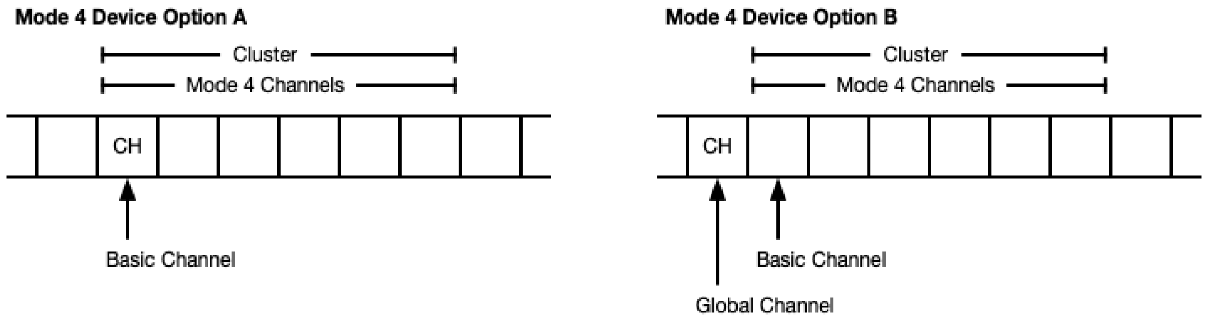

Figure 2 Mode 4 Device Option A and B Examples

Table 15 Mode 4 Device ChannelList Entry Option A and B

| | Mode 4 Option A | Mode 4 Option B |
|---|---|---|
| Header Data | `{"status":200}` | `{"status":200}` |
| Property Data | `[ { "title": "Mode 4 Device Option A", "channel": 3, "clusterChannelStart": 3, // Basic Channel "clusterLength": 6, "clusterMidiMode": 4, "programTitle": "Church Organ 1", "bankPC": [2,0,1] } ]` | `[ { "title": "Mode 4 Device Option B", "channel": 2, "clusterChannelStart": 3, // Basic Channel "clusterLength": 6, "clusterMidiMode": 4, "programTitle": "Church Organ 1", "bankPC": [2,0,1] } ]` |

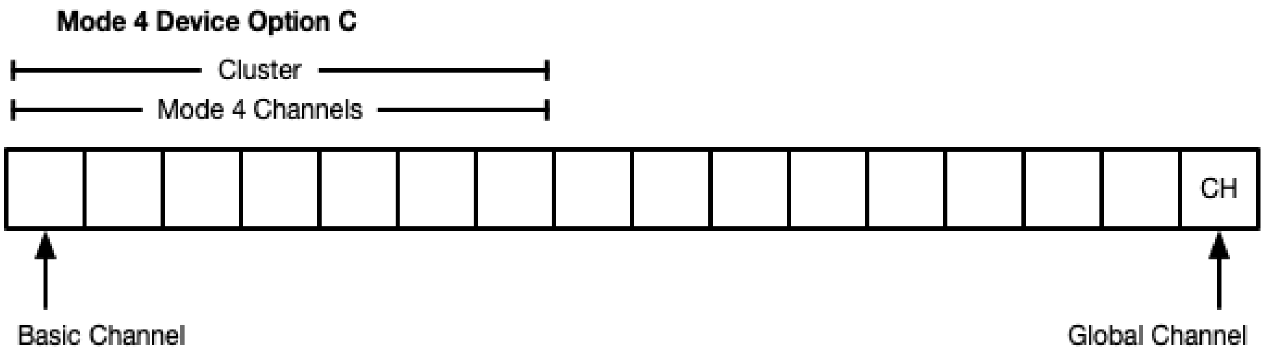

Figure 3 Mode 4 Device Option C Example

**Table 16 Property Data for a Mode 4 Device ChannelList Entry Option C**

| | |
|---|---|
| Header Data | {"status":200} |
| Property Data | [<br>  {<br>    "title": "Mode 4 Device Option C",<br>    "channel": 16,<br>    "clusterChannelStart": 1, // Basic Channel<br>    "clusterLength": 6,<br>    "clusterMidiMode": 4,<br>    "programTitle": "Guitar 1",<br>    "bankPC": [2,0,1]<br>   }<br>] |

### 4.6.3 MultiChannel Profile ChannelList Entry

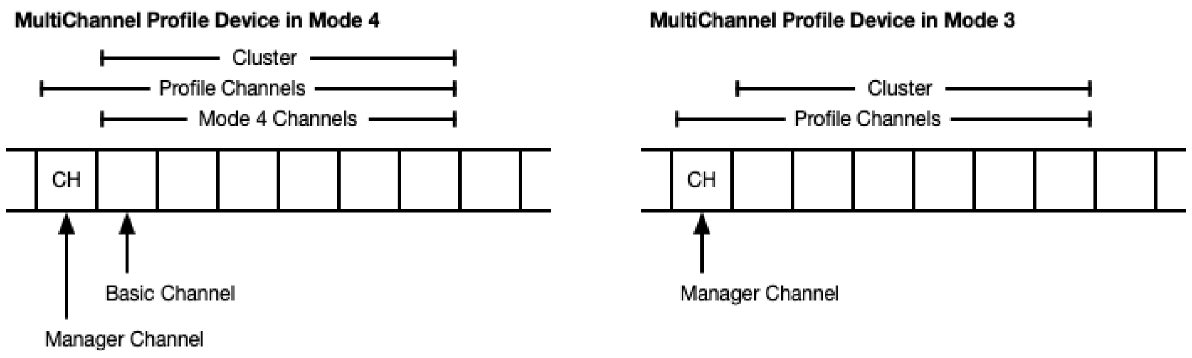

**Figure 4 MultiChannel Profile Examples**

**Table 17 MultiChannel Profile ChannelList Entry**

| | MultiChannel Profile Device in Mode 4 | MultiChannel Profile Device in Mode 3 |
|---|---|---|
| Header Data | {"status":200} | {"status":200} |
| Property Data | [<br>  {<br>    "title": "MultiChannel Profile Mode 4",<br>    "channel": 2,<br>    "clusterChannelStart": 3, // Basic Channel<br>    "clusterLength": 6,<br>    "clusterMidiMode": 4,<br>    "clusterType": "profile",<br>    "programTitle": "6 Manual Organ",<br>    "bankPC": [2,0,1],<br>   }<br>] | [<br>  {<br>    "title": MultiChannel Profile Mode 3",<br>    "channel": 2,<br>    "clusterChannelStart": 3, // Basic Channel<br>    "clusterLength": 6,<br>    "clusterType": "profile",<br>    "programTitle": "6 Manual Organ",<br>    "bankPC": [2,0,1],<br>   }<br>] |

### 4.6.4 MPE Device ChannelList Entry

MPE allows for one or two MPE setups per 16 Channels.

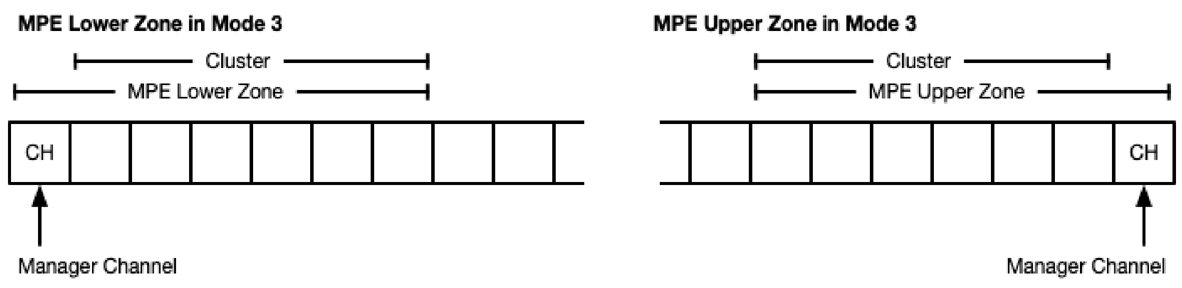

Figure 5 MPE Upper and Lower Zone in Mode 3 Examples

Table 18 MPE Upper and Lower Zone in Mode 3 ChannelList Entry

| | MPE Lower Zone in Mode 3 ChannelList Entry | MPE Upper Zone in Mode 3 ChannelList Entry |
|---|---|---|
| Header Data | `{"status":200}` | `{"status":200}` |
| Property Data | `[ { "title": "MPE Lower Zone Mode 3", "channel": 1, "clusterChannelStart": 2, "clusterLength": 6, "clusterType": "mpe1", "programTitle": "MPE Synth", "bankPC": [2,0,1] } ]` | `[ { "title": "MPE Upper Zone Mode 3", "channel": 16, "clusterChannelStart": 10, "clusterLength": 6, "clusterType": "mpe1", "programTitle": "MPE Synth", "bankPC": [2,0,1] } ]` |

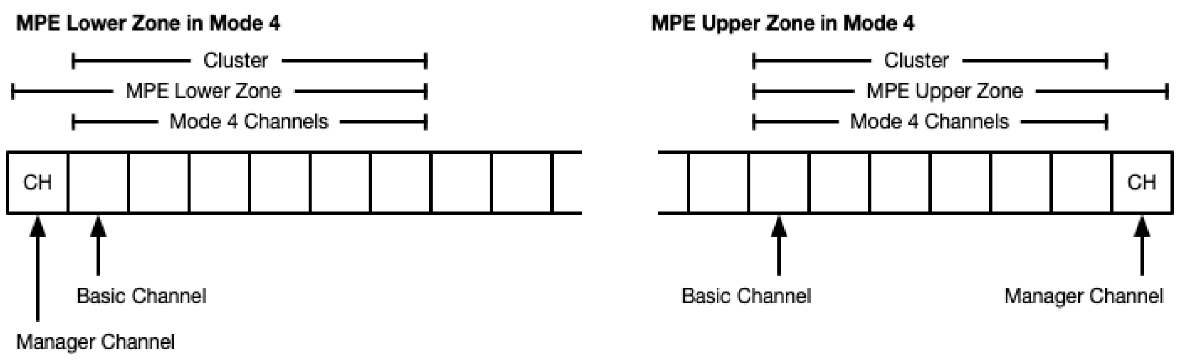

Figure 6 MPE Upper and Lower Zone in Mode 4 Examples

**Table 19 MPE Upper and Lower Zone in Mode 4 ChannelList Entry**

| | MPE Lower Zone in Mode 3 ChannelList Entry | MPE Upper Zone in Mode 3 ChannelList Entry |
|---|---|---|
| Header Data | {"status":200} | {"status":200} |
| Property Data | [<br>  {<br>    "title": "MPE Lower Zone Mode 4",<br>    "channel": 1,<br>    "clusterChannelStart": 2, // Basic Channel<br>    "clusterLength": 6,<br>    "clusterMidiMode": 4,<br>    "clusterType": "mpe1",<br>    "programTitle": "MPE Synth",<br>    "bankPC": [2,0,1],<br>   }<br>] | [<br>  {<br>    "title": "MPE Upper Zone Mode 4",<br>    "channel": 16,<br>    "clusterChannelStart": 10, // Basic Channel<br>    "clusterLength": 6,<br>    "clusterMidiMode": 4,<br>    "clusterType": "mpe1",<br>    "programTitle": "MPE Synth",<br>    "bankPC": [2,0,1],<br>   }<br>] |

## 4.6.5 Mode 1 and 2 Devices ChannelList Entry

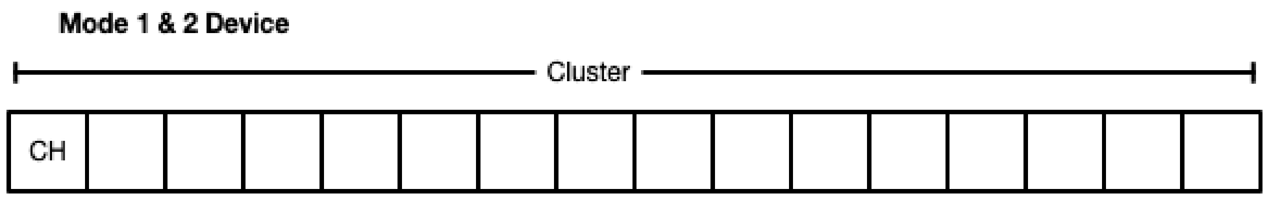

**Figure 7 Mode 1 and 2 Devices Example**

**Table 20 Mode 1 and 2 Devices ChannelList Entry**

| | |
|---|---|
| Header Data | {"status":200} |
| Property Data | [<br>  {<br>    "title": "Mode 1 Device",<br>    "channel": 1,<br>    "clusterChannelStart": 1, // Basic Channel<br>    "clusterLength": 16,<br>    "clusterMode": 1,<br>    "programTitle": "Omni Mode",<br>    "bankPC": [2,0,1],<br>   }<br>] |

## 4.7 ResourceList Integration for ChannelList

**Table 21 Minimal entry in ResourceList**

Property Data

```
[
    {"resource": "ChannelList"}
]
```

**Table 22 Full version with default settings**

Property Data

```
[
    {
        "resource": "ChannelList",
        "canGet": true,
        "canSet": "none",
        "canSubscribe": false,
        "canPaginate": false,
        "schema": {
            "type": "array",
            "title": "Channel List",
            "$ref": "http://schema.midi.org/property-exchange/M2-105-S_v1-
0_ChannelList.json"
        },
        "columns": [
            {"property": "title", "title": "Title"},
            {"property": "channel", "title": "MIDI Channel"},
            {"property": "programTitle", "title": "Program Title"},
            {"link": "ProgramList", "title": "Program List"}
        ]
    }
]
```

# 5 Foundational Resource: JSONSchema

The **JSONSchema** Resource provides the JSON Schema for Manufacturer Specific Resources **`"schema"`** property. See Common Rules for MIDI-CI Property Exchange ***[MA04]*** Section 12.4.2 for more information.

All Property Exchange Devices which implement Manufacturer Specific Resources which use the **`"$ref"`** Property in **ResourceList** schema with a Property Value in the form of `"midi+jsonschema://<JSON Schema Id>"` shall implement this **JSONSchema** Resource.

## 5.1 Initiator Requests Data from a Responder Using an Inquiry: Get Property Data

An Initiator may request the **JSONSchema** Resource from a Responder using an **Inquiry: Get Property Data** message.

Table 23 Initiator Sends Inquiry: Get Property Data

| | |
|---|---|
| `Header Data` | `{"resource":"JSONSchema","resId":"globalSchema"}` |
| `Property Data` | *none* |

*Note:* `"globalSchema"` *is a hypothetical example of Schemas as described in Common Rules for MIDI-CI Property Exchange. This would have been retrieved if this Property was declared in the* ***ResourceList*** *Property Data:* **`"$ref"`**`: "midi+jsonschema://globalSchema"`

Table 24 Responder Sends Reply to Get Property Data

| | |
|---|---|
| `Header Data` | `{"status":200}` |
| `Property Data` | see below |

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "type": "object",
    "properties": {
        "DeviceName": {
            "type": "string"
        },
        "audioOut": {
            "type": "string",
            "enum": ["USB","line"]
        },
        "midiThru": {
            "type": "boolean"
        }
    }
}
```

## 5.2 ResourceList Integration for JSONSchema

Table 25 Minimal entry in ResourceList

| | |
|---|---|
| `Property Data` | `[ {"resource": "JSONSchema"} ]` |

## Table 26 Full version with default settings

| | |
|---|---|
| Property Data | `[ { "resource": "JSONSchema", "canGet": true, "canSet": "none", "canSubscribe": false, "schema": { "type": "object", "title": "JSON Schema", "$ref": "http://json-schema.org/draft-04/schema" } } ]` |

http://www.amei.or.jp

https://www.midi.org