MIDI-CI Property Exchange

Foundational Resources:

DeviceInfo, ChannelList, JSONSchema

Version 1.01
November 24, 2020

Document M2-105-UM
Published By:
Association of Musical Electronics Industry
http://www.amei.or.jp
and
The MIDI Association
https://www.midi.org

MIDI™

PREFACE

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 [MMA02] and Common Rules for MIDI-CI Property Exchange [MMA03].

https://www.midi.org
http://www.amei.or.jp

AMEI Association of Musical Electronics Industry

MIDI™ Association

Table of Contents

1. Introduction

1.1 Background

Property Exchange is part of the MIDI Capability Inquiry (MIDI-CI) [MMA02] 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 [MMA02] and Common Rules for MIDI-CI Property Exchange specification [MMA03].

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

1.2 Related Documents

[MMA01] ***The Complete MIDI 1.0 Detailed Specification, Document Version 96.1, Third Edition***, Association of Musical Electronics Industry, http://www.amei.or.jp/, and MIDI Manufacturers Association, https://www.midi.org/.

[MMA02] ***MIDI Capability Inquiry (MIDI-CI), Version 1.1***, Association of Musical Electronics Industry, http://www.amei.or.jp/, and MIDI Manufacturers Association, https://www.midi.org/.

[MMA03] ***Common Rules for MIDI-CI Property Exchange, Version 1.1***, Association of Musical Electronics Industry, http://www.amei.or.jp/, and MIDI Manufacturers Association, https://www.midi.org/.

[MMA04] ***MIDI Polyphonic Expression, Version 1.0***, Association of Musical Electronics Industry, http://www.amei.or.jp/, and MIDI Manufacturers Association, https://www.midi.org/.

[MMA05] ***Universal MIDI Packet Format and MIDI 2.0 Protocol, Version 1.0***, Association of Musical Electronics Industry, http://www.amei.or.jp/, and MIDI Manufacturers Association, https://www.midi.org/.

[MMA06] ***MIDI-CI Property Exchange ProgramList Resource, Version 1.0***, Association of Musical Electronics Industry, http://www.amei.or.jp/, and MIDI Manufacturers Association, https://www.midi.org/.

1.3 Terminology

Basic Channel: When a Device is operating on multiple MIDI channels, the Basic Channel is one where MIDI messages can control parameters across multiple channels of the Device. For example, a Program Change sent on the Basic Channel could select sounds on multiple MIDI Channels of the Device.

Channel 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. A Channel Cluster may have a designated Basic Channel. Channels which constitute an MPE Zone are not a Channel Cluster.

Data Set: A complete Property Exchange message whether sent in one System Exclusive message in a single Chunk or in multiple Chunks.

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: One of 16 sets of MIDI data when using the Universal MIDI Packet Format. Each Group contains an independent set of System Messages, and 16 Channels that are equivalent to the MIDI 1.0 Protocol's 16 MIDI Channels. See [MMA05] Universal MIDI Packet Format and MIDI 2.0 Protocol.

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

MIDI 1.0 Specification: [MMA01] Complete MIDI 1.0 Detailed Specification, Document Version 96.1, Third Edition

MIDI-CI: [MMA02] MIDI Capability Inquiry.

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. MPE uses a defined note rotation mechanism which crosses multiple member Channels. Channels are used as a dynamic note-index, and Channel based controller messages are used as per-note-index controllers. Therefore, Channels which constitute an MPE Zone are not defined as a Channel Cluster. See [MMA04] MIDI Polyphonic Expression.

PE: Property Exchange.

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.

Property Data: A set of one or more Properties in a Device which are accessible by Property Exchange. Contained in the Property Data field of a MIDI-CI Property Exchange message.

Property Exchange: An AMEI/MMA specification, which is the basis for this specification, in which one Device may access Property Data from another Device.

Property Exchange Device: A Device which implements Property Exchange.

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.

Resource: A defined Property Data with an associated inquiry for accessing the Property Data.

Universal MIDI Packet: The data format of the Universal MIDI Packet (UMP), a data container for MIDI 1.0 Protocol messages and all MIDI 2.0 Protocol messages. See [MMA05] Universal MIDI Packet Format and MIDI 2.0 Protocol.

1.4 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 1 Words Relating to Specification Conformance

Word	Reserved For	Relation to Spec Conformance
shall	Statements of requirement	Mandatory. A conformant implementation conforms to all 'shall' statements.
should	Statements of recommendation	Recommended but not mandatory. 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. An implementation that does not conform to some or all 'may' statements is still conformant, providing 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 2 Words Not Relating to Specification Conformance

Word	Reserved For	Notes
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. 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. 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. 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. Not used for statements of conformance permission (see 'may' above).

2. Resources in this Specification

2.1 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 [MMA03] Common Rules for MIDI-CI Property Exchange).

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

- DeviceInfo

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

- ChannelList

This specification also defines one 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

3.1 Introduction

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. DeviceInfo also includes human-readable Properties for Manufacturer, Family, Model, Version information, and more.

All Property Exchange Devices shall implement this Resource.

3.2 Getting DeviceInfo Property Data

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

Initiator Sends Inquiry: Get Property Data Message

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

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

The object contains the following Properties:

Property Key	Property Value Type	Description
`manufacturerId`	array of numbers (integers), required	System Exclusive ID of the Device manufacturer in an array of 3 data byte values (as integers). For System Exclusive ID values that are only 1 byte in length, the System Exclusive ID value is the first value in the array and the remaining 2 values are filled with zeroes: [ID,0,0]
`familyId`	array of numbers (integers), required	This identifies the Device family, a related group of models to which the Device belongs, in an array of the 2 data byte values (as integers). The manufacturer is free to determine the grouping of models and the format of the data in this field. If the Responder does not have a familyID, then it uses [0,0].
`modelId`	array of numbers (integers), required	This identifies the Device family model number in an array of 2 data byte values (as integers). The manufacturer is free to determine the assignment of values and the format of the data in this field.
`versionId`	array of numbers (integers), required	This is the Software Revision Level of the Device in an array of 4 data byte values (as integers). This is typically the version of software or firmware but may also be version of hardware.
`manufacturer`	string, required	The human readable name of the Manufacturer.

family	string, required	The human readable product family name.
model	string, required	The human readable model name.
version	string, required	The human readable software/firmware/hardware version.
serialNumber	string	The Device should return a unique serial number. This is extremely valuable for uniquely identifying the Device, particularly in a system where 2 or more of the same model of Device might be connected.

Responder Sends Reply to Get Property Data Message

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],`
	`  "serialNumber": "12345678",`
	`  "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 [MMA03] Common Rules for MIDI-CI Property Exchange.

3.4 ResourceList Integration for DeviceInfo

Minimal entry in ResourceList:

Property Data	[{"resource": "DeviceInfo"}]

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

4.1 Introduction

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 Group of the Universal MIDI Packet Format. It describes the current Channel, Program, Group, MPE Status and other properties.

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

4.2 Getting ChannelList Property Data

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

Initiator Sends Inquiry: Get Property Data Message

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

Responder that supports ChannelList Resource shall return an array of objects in the Property Data using a Reply to Get Property Data Message.

Each object contains the following Properties:

Property Key	Property Value Type	Description
`title`	string, required	The name of the Channel being described
`channel`	number (integer) (1-16, required)	This is the Channel being described by this entry in the list of Channels.
`channelClusterId`	integer	If the Channel is a member of a Channel Cluster then this is the id of the Channel Cluster.
`clusterBasicChannel`	boolean	This declares if this is the Basic Channel for the Channel Cluster. If the Device is operating on multiple Channels which belong to the Channel Cluster, then MIDI messages sent to this Channel set some parameters across multiple Channels of the Channel Cluster.
`deviceBasicChannel`	boolean	This declares if this is the Basic Channel for the Device.
`programTitle`	string	The name of the Program currently active on this Channel.
`bankPC`	array of number (integer)s	This is a 3 item array containing the Bank MSB, Bank LSB and Program Change for the current Program.

mpeZone	enum ("upper", "lower")	If the Device is using MPE, then this Property declares if the Channel is part of the Upper Zone or Lower Zone.

Responder Sends Reply to Get Property Data Message

Header Data	{"status":200}

Property Data	[
	{
	"title": "Lead",
	"channel": 1,
	"programTitle": "Piano + Strings",
	"bankPC": [1,0,76]
	},
	{
	"title": "Drums",
	"channel": 10,
	"programTitle": "GM2 Jazz Drum Set",
	"bankPC": [0,0,33]
	}
	]

4.3 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 CMList, Manufacturer defined Resources, and others. See [MMA03] Common Rules for MIDI-CI Property Exchange.

4.3.1 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.

Property Data	[{ "title": "Lead", "channel": 1, "programTitle": "Piano + Strings", "bankPC": [1,0,76], "links":[{ "resource": "ProgramList", "resId": "general", "title":"General Programs" }, {"resource": "X-MidiEffects", "resId": "singch1"}]

	<pre> }]</pre>

4.3.2 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 [MMA06] MIDI-CI Property Exchange ProgramList Resource).

Property Data	<pre>[
	   {
	      "title": "Lead",
	      "channel": 1,
	      "programTitle": "Piano + Strings",
	      "bankPC": [1,0,76],
	      "links":[
	         {
	            "resource": "ProgramList", "resId": "presets",
	            "title": "Factory Presets"
	         },
	         {
	            "resource": "ProgramList", "resId": "user1",
	            "title": "User Programs"
	         },
	         {
	            "resource": "ProgramList", "resId": "gm2melodic",
	            "title": "GM2 Programs"
	         }
	      ]
	   },
	   {
	      "title": "Drums",
	      "channel": 10,
	      "programTitle": "GM2 Jazz Drum Set",
	      "bankPC": [0,0,33],
	      "links":[
	         {
	            "resource":"ProgramList", "resId":"gm2drums",
	            "title":"GM2 Drum Sets"
	         }
	      ]
	   }
	]</pre>

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.

4.4 Further Examples of ChannelList Property Data

4.4.1 Property Data for a Multi-Timbral Device that has no Basic Channel

A multichannel MIDI module has no Basic Channel.

Property Data

```
[
   {
      "title": "Lead",
      "channel": 1,
      "programTitle": "Acoustic Grand Piano",
      "bankPC": [0,0,1],
      "links":[
         {
            "resource": "ProgramList", "resId": "gmvoice",
            "title": "GM Voices"
         },
         {
            "resource": "ProgramList", "resId": "gmdrum",
            "title": "GM Drums"
         }
      ]
   },
   {
      "title": "Organ",
      "channel": 2,
      "programTitle": "Rock Organ",
      "bankPC": [0,0,1],
      "links":[
         {
            "resource": "ProgramList", "resId": "gmvoice",
            "title": "GM Voices"
         },
         {
            "resource": "ProgramList", "resId": "gmdrum",
            "title": "GM Drums"
         }
      ]
   },
   {
      "title": "Drums",
      "channel": 10,
      "programTitle": "Standard Kit",
      "bankPC": [120,0,1],
      "links":[
         {
            "resource": "ProgramList", "resId": "gmvoice",
            "title": "GM Voices"
```

	<pre>{
	"resource": "ProgramList", "resId": "gmdrum",
	"title": "GM Drums"
	}
	]
	}
	]</pre>

4.4.2 Response for a Multi-Channel Device with a Basic Channel

This may include Devices like a three-manual organ that may uses three different Channels for the upper manual, lower manual and bass pedals. In this example, Channel 16 is the Basic Channel, with "Masters" ProgramList which are Programs that select a set of sounds and parameters across all Channels of the Device.

Property Data	<pre>[
	{
	"programTitle": "Song 1",
	"channel": 16,
	"deviceBasicChannel": true,
	"title": "Master Channel",
	"bankPC": [0,0,21],
	"links":[
	{
	"resource": "ProgramList",
	"resId": "performance", "title": "Masters"
	}
	]
	},
	{
	"title": "Upper Swell",
	"channel": 1,
	"channelClusterId": 1,
	"clusterBasicChannel": true,
	"programTitle": "Hammond B3",
	"bankPC": [2,0,1],
	"links":[
	{
	"resource": "ProgramList", "resId": "organs",
	"title": "Organs"
	}
	]
	},
	{
	"title": "Foot Pedals",
	"channel": 3,</pre>

```
      "channelClusterId": 1
   },
   {
      "title": "Lower Swell",
      "channel": 2,
      "channelClusterId": 1
   },
   {
      "title": "Piano",
      "channel": 4,
      "programTitle": "Honky Tonk 2",
      "bankPC": [3,0,1],
      "links":[
         {
            "resource": "ProgramList", "resId": "pianos",
            "title": "Pianos"
         }
      ]
   },
   {
      "title": "Bass",
      "channel": 5,
      "programTitle": "Bass Synth",
      "bankPC": [4,0,30],
      "links":[
         {
            "resource": "ProgramList", "resId": "synths",
            "title": "Synths"
         }
      ]
   },
   {
      "title": "Drums",
      "channel": 10,
      "programTitle": "Rock Kit",
      "bankPC": [120,0,20],
      "links":[
         {
            "resource": "ProgramList", "resId": "drums",
            "title": "Drums"
         }
      ]
   }
]
```

4.4.3 Response for an MPE Device

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

Property Data	

```
[
   {
      "title": "MPE Master Channel",
      "channel": 1,
      "programTitle": "Super Synth",
      "bankPC": [0,0,1],
      "mpeZone": "lower",
      "links":[
         {
            "resource": "ProgramList",
            "resId": "mpepatches", "title": "MPE Programs"
         }
      ]
   },
   {"title": "Member Channel", "channel": 2, "mpeZone": "lower"},
   {"title": "Member Channel", "channel": 3, "mpeZone": "lower"},
   {"title": "Member Channel", "channel": 4, "mpeZone": "lower"},
   {"title": "Member Channel", "channel": 5, "mpeZone": "lower"},
   {"title": "Member Channel", "channel": 6, "mpeZone": "lower"},
   {"title": "Member Channel", "channel": 7, "mpeZone": "lower"},
   {"title": "Member Channel", "channel": 8, "mpeZone": "lower"}
]
```

4.5 ResourceList Integration for ChannelList

Minimal entry in ResourceList:

Property Data	

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

Full version with default settings:

Property Data	

```
[
   {
      "resource": "ChannelList",
      "canGet": true,
      "canSet": "none",
      "canSubscribe": false,
      "canPaginate": false,
      "schema":{
```

	<pre>"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"}]
	}
	]</pre>

5. Foundational Resource: JSONSchema

5.1 Introduction

The "JSONSchema" Resource provides the JSON Schema for Manufacturer Specific Resources "schema" property. See [MMA03] Common Rules for MIDI-CI Property Exchange 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.

5.2 Getting JSONSchema Property Data

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

Initiator Sends Inquiry: Get Property Data Message

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"

Responder Sends Reply to Get Property Data Message

Header Data	`{"status":200}`

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

5.3 ResourceList Integration for JSONSchema

Minimal entry in ResourceList:

Property Data	[{"resource": "JSONSchema"}]

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"
	  }
	  }
	]

Revision History

Date	Version	Changes
Nov. 17, 2020	1.01	Initial Version

https://www.midi.org
http://www.amei.or.jp