MCPC TR-007
Bluetooth Audio
Technical Reference
Version 1.0
Revision 2

February. 9th, 2007

Mobile Computing Promotion Consortium

(MCPC)

Revision History

Version	Date	Description	Details
	2005/04/15		New Document
V0.50	2005/06/14	All	Review
V0.51	2005/07/05	All	Translation.
V0.60	2005/10/11	All	
V0.70	2005/11/07	All	
V0.71	2005/11/28	All	
V0.80Draft	2006/01/17	All	
V0.89Draft1	2006/02/16		
V0.89Draft2	2006/02/19		
V0.89Draft3	2006/02/21		
D0.90r1	2006/02/22		Change numbering of document following to SIG
D0.90r2	2006/02/23		
D0.90r4	2006/03/07		
D0.90r5	2006/03/08		
D0.90r6	2006/03/13		Obtaining Company ID
D1.00r0	2006/03/14		Fixing all issues.
V1.00	2006/03/28		Released
V1.00r1			Modify following SIG AVRCP V.1.3 as below. “Device Settings” is replaced with “Player Application Settings”. “GetPlayStatus” PDU ID is moved to “Notification PDUs” from “Media Information PDUs”, and is changed PDU ID to 0x30 from 0x21. PDU ID of “RegisterNotification” is replaced with 0x31. PDU ID of “SuspendNotify” is replaced with 0x32. PDU ID of “ResumeNotify” is replaced with 0x33. “5.6. PASS THROUGH COMMAND” is renamed to “Basic Group Navigation”, and the contents are modified. “5.1 Configuration PDUs” is replaced with “5.1 Capabilities PDUs”. Modify “Table 4.2.1 List of command”. Modify “Table 4.2.2 List of PASS THROUGH COMMAND”.
V1.00r2	2007/2/7		- Reflect a document of SIG AVRCP D16r03 to this document. - Remove command of SuspendNotify(5.4.3) - Remove command of ResumeNotify(5.4.4) - Modify “5.1.1 GetCapabilities” - Added Appendix 4 - minor editorial changes.

Contributors

Masahiko Seki	Sony Corporation
Haruhiko Kaneko	Sony Corporation
Mitsuyoshi Yasuda	Sony Ericsson Mobile Communications Japan,
Akio Konishi	FUJITSU DEVICES INC.,
Masahiko Nakashima	FUJITSU DEVICES INC.,
Katsutoshi Arata	FUJITSU TEN LIMITED
Kenji Kagami	FUJITSU TEN LIMITED
Soichi Saito	DENSO CORPORATION
Fusako Inotsume	Pioneer Corporation
Issei Sugawara	Clarion Co., Ltd.
Seiichi Suzuki	Clarion Co., Ltd.
Hiroyuki Morimoto	Mitsubishi Electric Corporation
Kazuhiko Sawaki	Johnson Control Inc.,
Hiroshi Matsuya	TOSHIBA CORPORATION
Makoto Yamashita	TOSHIBA CORPORATION
Shuichi Sakurai	TOSHIBA CORPORATION
Baskar S.	Impulsesoft
K. A. Srinivasan	Impulsesoft
Ankit Jain	Impulsesoft

MCPC Bluetooth Audio Sub Working Group Members (Alphabetical Order)

ALPINE ELECTRONICS, INC.	Nobuki Asai
ALPS ELECTRIC CO., LTD.	Masanobu Ujiie, Talguk Kim
Clarion Co., Ltd.	Issei Sugawara, Seiichi Suzuki
DENSO CORPORATION	Hideaki Suzuki, Masashi Miura, Soichi Saito
ELECOM CO., LTD.	Shuji Terakado
e-w/you Inc.	Shunsuke Matsuda, Y. Ohkubo
FUJITSU LIMITED	Yoshihiro Takamatsuya
FUJITSU DEVICES INC.,	Akio Konishi, Masahiko Nakashima
FUJITSU TEN LIMITED	Katsutoshi Arata, Yoshitaka Hirashima, Kenji Kagami
Hitachi, Ltd.	Motohisa Higuchi
Impulsesoft Private Limited.	Baskar S., K. A. Srinivasan, Ankit Jain
Japan Automobile Manufacturers Association, Inc. (JAMA)	K. Akahori
Johnson Controls, Inc.	Kazuhiko Sawaki
KDDI CORPORATION	J. Ikeno, Y. Yamada
MITSUBISHI ELECTRIC CORPORATION	Hiroyuki Morimoto, Riko Yagiu, Shuichi Nishikawa, Takanori Niwa
MITSUMI ELECTRIC CO., LTD.	H. Abe, Hiroshi Fujikake,
Mobilecast Inc.	K. Nakamori, Naoya Funakoshi
Murata Manufacturing Co., Ltd.	Kazuhiro Okada
NISSAN MOTOR CO., LTD. Representative for JAMA	Stephane Bouet, Tomoyuki Suzuki, Toshiki Haga
NTT DoCoMo, Inc.	Koichi Sawai, Kosuke Kakuno, Toru Tanaka
Open Interface Inc.	Shunsuke Nakajima, Yasuhiro Sakaguchi
Panasonic Mobile Communications Co., Ltd.	Akinori Ohta
Pioneer Corporation	Fusako Inotsume
SANYO Electric Co., Ltd.	Hiroki Seyama
Sony Corporation	Haruhiko Kaneko, Masahiko Seki
Sony Ericsson Mobile Communications Japan, Inc.	Mitsuyoshi Yasuda
TAIYO YUDEN CO., LTD.	Daigo Kobayashi, Hiroshi Kato, Terumichi Nakashima, Youjiro Shimizu
TOSHIBA CORPORATION	Hiroshi Matsuya, Makoto Yamashita, Shuichi Sakurai
Vodafone K.K.	Satoshi Miyazaki, Shio Kubo

Document Terminology

The MCPC has adopted Section 13.1 of the IEEE Standards Style Manual, which dictates use of the words “shall”, “should”, “may” and “can” in the development of documentation, as follows.

- The word ***shall*** is used to indicate mandatory requirements strictly to be followed in order to conform to the standard and from which no deviation is permitted (***shall*** equals is ***required to***).

- The use of the word ***must*** is deprecated and shall not be used when stating mandatory requirements; ***must*** is used only to describe unavoidable situations.

- The use of the word ***will*** is deprecated and shall not be used when starting mandatory requirements; ***will*** is only used in statements of fact.

- The word ***should*** is used to indicate that among several possibilities on is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain course of action is deprecated but not prohibited (***should*** equals ***is recommended that***).

- The word ***may*** is used to indicate a course of action permissible within the limits of the standard (***may*** equals ***is permitted***).

- The word ***can*** is used for statements of possibility and capability, whether material, physical or causal (***can*** equals ***is able to***).

Content

1 Introduction

1.1. Scope

A demand for wireless device has been increasing along with the growth of portable music players in the market.

Wireless device is specialized for remote controlled function in AV profile that is regulated by Bluetooth wireless communication technology.

However, most of wireless devices in the market have representational functions at remote controller to display music.

If specification for displaying or controlling application of AV profile is different from one to another, it loses commonality of Bluetooth technology enabled device.

Also, MCPC holds concern about connecting of each product, which is manufactured by each manufacturer, cannot be ensured enough.

For above, MCPC has defined a technical reference to transfer the music data to be more convenient for users using Bluetooth Technology. It gives rise to an improvement in connectivity of products such as portable music player and car-mounted devices (e.g. Headphone). In addition, it enables to support users to implement AV profile correctly for the following items.

- Recommended sequence
- Recommended parameter of commands
- How to use vendor's optional parts

This technical reference is applied to an application for AV PROFILE.
And Basic Commands and sequence is based on "AVRCP V.1.0 Adapted" which is published by Bluetooth SIG Inc,.

[Note]
The contents of this document shall be non-binding, and shall not infringe any rights of Bluetooth®. The use of this document or any of the contents thereof shall be made at the responsibility of each user. MCPC makes no warranties whatsoever, whether expressed or implied, with regard to any matters relating to this document or contents thereof, including, without limitation, any infringement of legal right, marketability, or fitness for a particular purpose.

1.2. System

Following Figure 1 shows an example of system configuration defined by this technical reference.

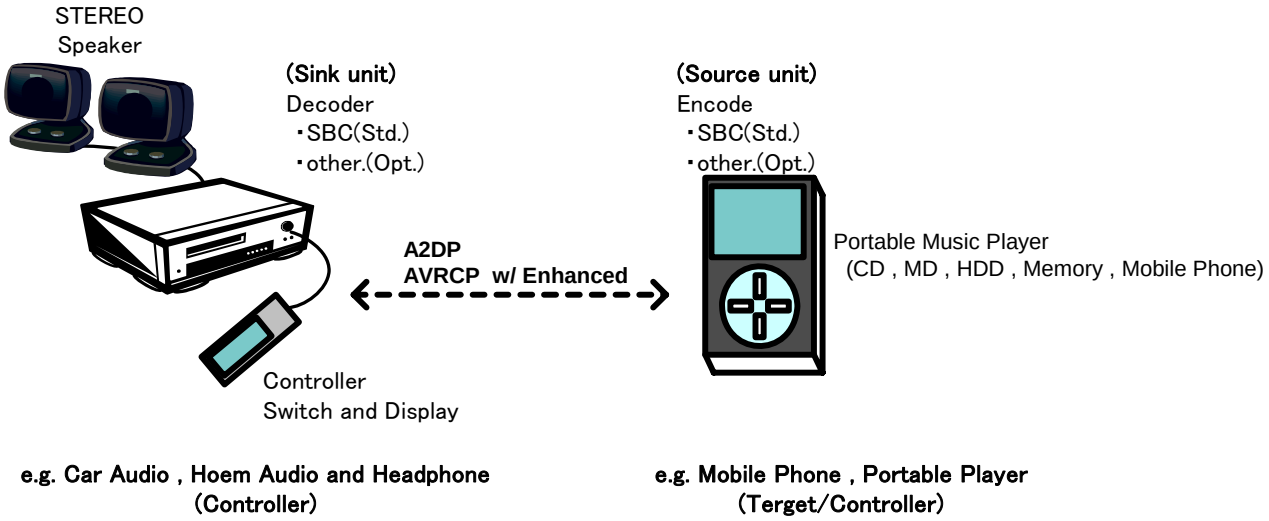

Figure 1 example system configuration

1.3. AVRCP Definition

- Car audio/Home audio/Headphone/Mobile Phone are CT, and Mobile Phone/Portable audio players are TG.
- CT supports operation_id that corresponds to Category 1(Player/Recorder).
  **Does not support Recorder.
- Moreover, it is required to expand function depending on VENDOR DEPENDENT Command or PASS THROUGH Command (VENDOR UNIQUE).
- MCPC applies for Vendor ID. ID owned by each vendor is defined by VENDOR DEPENDENT.

1.4. Basic concept

- The purpose is to expand the area that cannot be covered by PASS THROUGH command / response of AVRCP.
- Expand without changing the basic part of AVRCP. Thus, CT is assigned to control, and TG is assigned to be controlled as ROLE.
- Assume following items to be connected.
  CT: Car audio / Home Audio / AV Headphone / Mobile Phone etc
  TG: Mobile Phone / Portable audio player etc.
- Use Vendor Unique of PASS THROUGH command/response for key undefined by PASS THROUGH command / response of AVRCP.
- Use VENDOR DEPENDENT command / response to send / receive data(Information of meta and status etc.).
- TG gives a response as corresponding to command given by CT. This is considered as a basic operation for sequence. If change in the status is made and need to notify it from TG actively, use NOTIFY command of AVRCP to register for issuing a defined NOTIFY command tells the change from CT. When NOTIFY command is issued, status is sent to CT with the use of INTERIM response from TG, CHANGED response is used to notify the change made in the status later.
- Considering the case which VENDOR DEPENDENT command/response cannot stored in MTU (512 bytes) of AVRCP in one time, MCPC provides for mechanism to continue sending command / response, and to receive / send large data.
- Following an AVRCP disconnection, CT and TG shall not assume that any state from the previous connection is valid (such as Player Application Setting, Character Set, RegisterNotification).
- **When you develop Bluetooth wireless technology enabled devices/products using this Technical Reference, you should refer the latest version of Bluetooth SIG published document during your development and before you release your devices/products to Market.**
- **The "Company ID", which is appearing in this documentation, is for this Technical Reference, not same as "Company ID" for Bluetooth SIG.**

2. Usage Model

2.1. Basic usage model.

The basic usage model is shown in Figure 2.

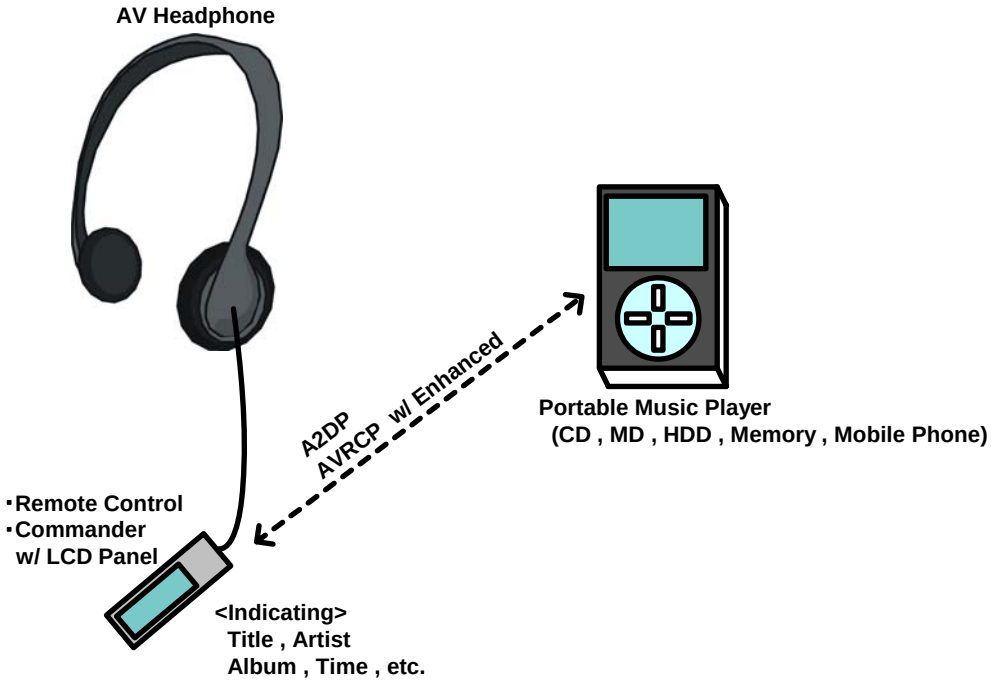

Figure 2 Basic usage model

The CT is a headphone with LCD display and the TG is a portable audio player like as CD, HDD, Memory, and Mobile Phone.

The user can listen to the music with the headphone through A2DP connection and can control the portable audio player through standard AVRCP connection.
In addition, the user also can see some information on the LCD display through enhanced AVRCP connection that is defined in this document.

Types of information that can be displayed are Track Name, Artist Name, Album Name, Time, etc.

3. Data Representation

3.1. Transfer Byte Order

Multiple-byte fields are transferred in standard network byte order (Big Endian), with more significant (high-order) bytes being transferred before less-significant (low-order) bytes.

3.2. Protocol Data Unit Format

The vendor_dependent_data field in the vendor dependent command/response frames will be a PDU (Protocol Data Unit). Every PDU consists of a PDU Identifier, length of all parameters (excluding the parameter length field) and the PDU-specific parameters.

Oct	MSB (7)	6	5	4	3	2	1	LSB (0)
0	PDU ID							
1	Reserved						Packet Type	
2-3	Parameter Length							
4-n	Parameter (Number of parameters determined by Parameter Length)							

PDU format

The PDU fields are briefly described below:

PDU ID: The PDU ID is used to identify the specific command/response with unique identifier for each operation.

Packet Type: The Packet Type field qualifies each packet as either first (Packet Type=01), continue (Packet Type=10), or end packet (Packet Type=11). In the case of a non-fragmented message, this field (Packet Type=00) simply indicates that the message is sent in a single AV/C frame.

The packets are fragmented by the sender so as to be able to accommodate into the 512 bytes AV/C packet size restriction. Receivers have the flexibility to request continuation packets at their convenience from the sender or abort the continuation request. Note: if the L2CAP MTU is less than 512 bytes, AVCTP will also apply fragmentation to each of the AV/C packets. All response fragments shall have the same PDU ID as the original request.

A sender shall not interleave fragmented PDUs. Once a sender has sent a start fragment it shall only send further fragments of that PDU until that PDU is completed or aborted. If a receiver receives a start fragment or non-fragmented Metadata-Transfer message when it already has an incomplete fragment from that sender then the receiver shall consider the first PDU aborted. A PASSTHROUGH command may be interleaved in fragmented Metadata-Transfer communication without aborting it.

Parameter Length: The parameter Length field specifies the length of the all the parameters following the Parameter Length field.

Parameter1 …n: These are the parameters for the specific operations performed and are described in sections below.

An example command (GetCapabilities) PDU will be as follows:

Oct	MSB (7)	6	5	4	3	2	1	LSB (0)
0	0x0				Ctype: 0x1 (STATUS)			
1	Subunit_type:0x9 (PANEL)					Subunit_ID: 0x0		
2	Opcode: 0x0 (VENDOR DEPENDENT)							
3 – 5	Company ID: 0x0017A7 MCPC registered CompanyID							
6	PDU ID (0x10 - Get Capabilities)							
7	Reserved (0x00)						Packet Type (0x0)	
8 – 9	Parameter Length (0x0001)							
10	Capability ID (0x1 - PROFILE VERSION)							

Table 3.2.1 An example command (GetCapabilities) PDU

The grayed portion in table above indicates the PDU inside an AV/C Vendor dependent command frame.

3.3. PASS THROUGH command (vendor unique)

PASS THROUGH Command (vendor unique) Frame

Oct	MSB (7)	6	5	4	3	2	1	LSB (0)
0	0x0				Ctype: 0x0 (CONTROL)			
1	Subunit_type:0x9 (PANEL)					Subunit_ID: 0x0		
2	Opcode: 0x7C (PASS THROUGH)							
3	State_flag *2	Operation_ID: 0x7E (VENDOR UNIQUE)						
4	Operation_data_field_length: 0x5							
5 - 7	Company ID: 0x0017A7							
8 - 9	Vendor_unique_id							

Table 3.3.1 PASS THROUGH Command Frame

PASS THROUGH Response (vendor unique) Frame

Oct	MSB (7)	6	5	4	3	2	1	LSB (0)
0	0x0				Response *1			
1	Subunit_type:0x9 (PANEL)					Subunit_ID: 0x0		
2	Opcode: 0x7C (PASS THROUGH)							
3	State_flag *2	Operation_ID: 0x7E (VENDOR UNIQUE)						
4	Operation_data_field_length: 0x5							
5 - 7	Company ID: 0x0017A7							
8 - 9	Vendor_unique_id							

Table 3.3.2 PASS THROUGH Response Frame

*1 0x8(NOT_IMPLEMENTED), 0x9 (ACCEPTED), 0xA (REJECTED)

*2 A CT sends a command frame with its state_flag field in value 0 when a button is pushed and in value 1 when the button is released.

3.4. VENDOR-DEPENDENT Command

(VENDOR DEPENDENT Command Frame)

Oct	MSB (7)	6	5	4	3	2	1	LSB (0)
0	0x0				Ctype : *1			
1	Subunit_type:0x9 (PANEL)					Subunit_ID: 0x0		
2	Opcode: 0x0 (VENDOR－DEPENDENT)							
3 – 5	Company ID: 0x0017A7							
6	PDU ID							
7	Reserved (0x0)						P_type : *2	
8 – 9	Parameter Length							
10 – n	Parameter(s)							

*1 0x0 (CONTROL), 0x1 (STATUS), 0x3 (NOTIFY)

*2 Packet type : 00(Non_Fragmented), 01(First), 10(Continue), 11(End)

Table 3.4.1 VENDOR DEPENDENT Command Frame

(VENDOR DEPENDENT Response Frame)

Oct	MSB (7)	6	5	4	3	2	1	LSB (0)
0	0x0				Response: *3			
1	Subunit_type:0x9 (PANEL)					Subunit_ID: 0x0		
2	Opcode: 0x0 (VENDOR－DEPENDENT)							
3 – 5	Company ID: 0x0017A7							
6	PDU ID							
7	Reserved (0x0)						P_type : *2	
8 – 9	Parameter Length							
10 – n	Parameter(s)							

*3 0x8 (NOT_IMPLEMENTED), 0x9 (ACCEPTED), 0xA (REJECTED), 0xC (STABLE), 0xD (CHANGED), 0xF (INTERIM)

Table 3.4.2 VENDOR DEPENDENT Response Frame

4. List of command for support

4.1. Feature Support

The table below shows the feature requirements for this profile. Note that a device may have both CT and TG capabilities. In that case, features for both CT and TG are required.

	Feature	Support in CT	Support in TG
1.	Connection establishment for control	M	O
2.	Release connection for control	M	M
3.	Sending UNIT INFO command	O	X
4.	Receiving UNIT INFO command	X	M
5.	Sending SUBUNIT INFO command	O	X
6.	Receiving SUBUNIT INFO command	X	M
7.	Sending VENDOR DEPENDENT command	C*[3]	X
8.	Receiving VENDOR DEPENDENT command	X	C*[3]
9.	Sending PASS THROUGH command	M	X
10.	Receiving PASS THROUGH command	X	M
11.	**Capabilities**	O	C*[1]
12.	**Player Application Settings**	O	O
13.	**Metadata Attributes**	O	C*[1]
14.	**Notifications**	C*[2]	C*[2]
15.	**Continuation**	C*[2]	C*[2]
16.	**Basic Group Navigation**	O	O

Table 4.1.1 Application Layer Features

C*[1] – Mandatory if Target supports Category 1 or optional otherwise
C*[2] – Mandatory if Controller supports Metadata Attributes or optional otherwise
C*[3] – Mandatory if any of 11,12,13, 14, 15 features are supported or optional otherwise

* Note that sending and receiving VENDOR DEPENDENT commands are Option in original AVRCP document. However some VENDOR DEPENDENT commands in this document shall be Mandatory so that these supports also shall be Mandatory.

4.2. List of commands for AVRCP Enhancement

List of commands are given in Table 4.2.1

Section	PDU ID	PDU Support in CT support in TG Commands	AV/C Command Type	CT	TG	TG Response Time
5.1		**Capabilities**				
5.1.1	0x10	GetCapabilities	STATUS	M	M	T_{MTP}
5.2		**Player Application Setting**				
5.2.1	0x11	ListPlayerApplicationSettingAttributes	STATUS	M	M	T_{MTP}
5.2.2	0x12	ListPlayerApplicationSettingValues	STATUS	O	M	T_{MTP}
5.2.3	0x13	GetCurrentPlayerApplicationSettingValue	STATUS	C*2	M	T_{MTP}
5.2.4	0x14	SetPlayerApplicationSettingValue	CONTROL	C*2	M	T_{MTC}
5.2.5	0x15	GetPlayerApplicationSettingAttributeText	STATUS	O	C*1	T_{MTP}
5.2.6	0x16	GetPlayerApplicationSettingValueText	STATUS	O	C*1	T_{MTP}
5.2.7	0x17	InformDisplayableCharacterSet	CONTROL	O	O	T_{MTC}
5.2.8	0x18	InformBatteryStatusOfCT	CONTROL	O	O	T_{MTC}
5.3		**Media Attributes**				
5.3.1	0x20	GetElementAttributes	STATUS	M	M	T_{MTP}
5.4		**Notification**				
5.4.1	0x30	GetPlayStatus	STATUS	O	M	T_{MTP}
5.4.2	0x31	RegisterNotification	NOTIFY	M	M	T_{MTP}
5.5		**Continuing**				
5.5.1	0x40	RequestContinuingResponse	CONTROL	M	M	T_{MTC}
5.5.2	0x41	AbortContinuingResponse	CONTROL	M	M	T_{MTC}

Table 4.2.1 List of command

Notes

C*1 If player application setting attribute IDs for menu extension (Refer to Appendix 2) are supported then mandatory or optional otherwise.

C*2 Either Get or Set player application settings shall be mandatory.

Requirements for CT refer to the ability to send a command.
Requirements for TG refer to the ability to respond to a command. The AV/C command type of the response PDU shall be per the AV/C specification's definitions for responses.

For error response PDU the response parameter is always the error code independent of the response format defined for ACCEPTED PDU response for the corresponding PDU command.

All strings passed in Technical reference PDUs are not null terminated.

T_{MTC} is the time period before which TG is expected to generate a response frame for CONTROL commands. (Refer to Appendix 5)
T_{MTP} is the time period before which TG is expected to generate a response frame for interim response for NOTIFY commands and final response for STATUS commands. (Refer to Appendix 5)

The Technical reference feature adds the following operations that shall be used with PASS THROUGH command:

Section	Vendor Unique ID	Operation Name	AV/C Command Type	CT	TG
5.6		**Basic Group Navigation**			
5.6.1	0x0000	Next Group	CONTROL	M	M
5.6.2	0x0001	Previous Group	CONTROL	M	M

Table 4.2.2 List of PASS THROUGH COMMAND

These PASS THROUGH commands shall use MCPC registered CompanyId as the opcode with the defined vendor unique Id with the PANEL subunit-type.

Requirements for CT refer to the ability to send a command.
Requirements for TG refer to the ability to respond to a command.

5. Function of each command and parameter

5.1. Capabilities PDUs

5.1.1. GetCapabilities (PDU ID: 0x10)

Description:

This primitive gets the capabilities supported by remote device. This could be sent by CT to inquire capabilities of the peer device.

Command format (GetCapabilities)

Parameters	Size(byte)	Description	Allowed Values
CapabilityID	1	Specific capability requested.	Defined CapabilityIDs are: PROFILE_VERSION (0x1): This requests the profile version supported by remote device COMPANY_ID (0x2): This requests the list of CompanyID supported by TG. EVENTS_SUPPORTED (0x3): This requests the list of events supported by the TG. TG is expected to respond with all the events supported including the mandatory events defined in this specification. Other CapabilityIDs are reserved

GetCapabilities Response format for PROFILE_VERSION

Parameters	Size(byte)	Description	Allowed Values
capabilityID	1	Specific capability requested	PROFILE_VERSION
CapabilityCount (n)	1	Specifies the length of capability value returned for a CapabilityID	
Capability	1…n	Capability provided	Current profile version supported is "1_00"

GetCapabilities Response for COMPANY_ID

Parameters	Size(byte)	Description	Allowed Values
capabilityID	1	Specific capability requested	COMPANY_ID
CapabilityCount (n)	1	Specifies the number of COMPANY_ID returned	1-255
Capability	3…3*n	List of COMPANY_ID	CompanyID value range as defined in AV/C VENDOR_DEPENDENT command frame. Each CompanyID is 3 bytes long.

GetCapabilities Response for EVENTS_SUPPORTED

Parameters	Size(byte)	Description	Allowed Values
capabilityID	1	Specific capability requested	EVENTS_SUPPORTED
CapabilityCount (n)	1	Specifies the number of events supported returned	2-255
Capability	2…n	List of EventsIDs	Minimum of two mandatory EventIDs defined in Appendix 4 is returned. EventIDs are 1 byte each.

NOTE: The CT should be aware that the capabilities supported by the TG may be subject to change. This may occur if the application on the TG changes, or the application changes mode, for instance different functionality may be available when the TG is playing locally stored audio tracks to when it is acting as a radio. How this is handled by the CT is implementation dependent. If the TG application changes to support less functionality the CT may receive error responses indicating that the function requested is not implemented. The CT may then decide to reissue the GetCapabilities to get the most current capabilities. If the TG application changes to support more features the CT may be happy to continue using the original set of features supported. If not it may choose to occasionally poll the TG with a GetCapabilities to determine when further capabilities are available.

5.2. Player Application Settings PDUs

The player application settings PDU provides mechanism for controller devices to query player application setting attributes, get and set specific setting value for these attributes.

It is assumed that all player application settings are available on the target as an <attribute, value> pair. It is assumed for each device attribute there shall be multiple possible values, with one of them being the current set value.

The specification defines pre-defined attributes and values for some of the commonly used player application settings. The PDUs allow for extensions to the pre-defined attributes and values defined in the target and are accessible to the controller, along with displayable text. This will allow controllers without the semantic understanding of the target's player application setting to be able to display setting related text and provide users mechanism to operate on the player application settings.

Each player application setting has a unique AttributeID and the attributes have values that have a valueID. Target-defined attributes and values have displayable text associated with them for allowing controllers to query for this information.

The following PDUs provide the needed functionality for controller devices to access and set attribute value on the target device.

5.2.1. ListPlayerApplicationSettingAttributes (PDU ID: 0x11)

Description:

This primitive request the target device to provide target supported player application setting attributes. The list of reserved player application setting attributes is provided in Appendix 2. It is expected that a target device may have additional attributes not defined as part of this technical reference.

Command format (ListPlayerApplicationSettingAttributes)

Parameters	Size(byte)	Description	Allowed Values
None			

Response format (ListPlayerApplicationSettingAttributes)

Parameters	Size(byte)	Description	Allowed Values
NumPlayerApplicationSettingAttributes(N)	1	Number of attributes provided	0-255
PlayerApplicationSettingAttributeID	1	Specifies the player application setting attribute ID	See Appendix 2 for the list of Player Application Setting attribute IDs

And so on for the number of target defined player application setting attributes (N).

5.2.2. ListPlayerApplicationSettingValues (PDU ID: 0x12)

Description:

This primitive requests the target device to list the set of possible values for the requested player application setting attribute. The list of reserved player application setting attributes and their values are provided in Appendix 2. It is expected that a target device may have additional attribute values not defined as part of this technical reference.

Command format (ListPlayerApplicationSettingValues)

Parameters	Size(byte)	Description	Allowed Values
PlayerApplicationSettingAttributeID	1	Player application setting attribute ID	Player application setting attribute ID as defined in Appendix 2 or received from the target

Response format (ListPlayerApplicationSettingValues)

Parameters	Size(byte)	Description	Allowed Values
NumPlayerApplicationSettingValues (N)	1	Number of player application setting values	1-255
PlayerApplicationSettingValueID1	1	Specifies the player application setting value ID	See Appendix 2 for list of reserved player application setting values. Additional values may be provided by the target for the requested player application setting attribute

And so on for the number of target defined player application setting values (N).

5.2.3. GetCurrentPlayerApplicationSettingValue (PDU ID: 0x13)

Description:

This primitive requests the target device to provide the current set values on the target for the provided player application setting attributes list.

Command format (GetCurrentPlayerApplicationSettingValue)

Parameters	Size(byte)	Description	Allowed Values
NumPlayerApplicationSettingAttributeID (N)	1	Number of player application setting attribute for which current set values are requested	1-255
PlayerApplicationSettingAttributeID1	1	Player application setting attribute ID for which the corresponding current set value is requested	Valid PlayerApplicationSettingAttributeID values received from the target, or defined as part of Appendix 2

And so on for the number of target defined player application setting attributes in the requested order (N).

Response format (GetCurrentPlayerApplicationSettingValue)

Parameters	Size(byte)	Description	Allowed Values
NumPlayerApplicationSettingValues (N)	1	Number of player application settings value provided	1-255
PlayerApplicationSettingAttributeID1	1	Player application setting attribute ID for which the value is returned	1-255
PlayerApplicationSettingValueID1	1	Currently set player application setting value on the target for the corresponding requested player application setting attribute ID	Valid PlayerApplicationSettingValueID values received from the target, or defined as part of Appendix 2

And so on for the number of target defined player application setting values in the requested order (N).

5.2.4. SetPlayerApplicationSettingValue (PDU ID: 0x14)

Description:

This primitive requests to set the player application setting list of player application setting values on the target device for the corresponding defined list of PlayerApplicationSettingAttributes.

Command format (SetPlayerApplicationSettingValue)

Parameters	Size(byte)	Description	Allowed Values
NumPlayerApplicationSettingAttributes (N)	1	Number of player application setting attributes for which the player application setting	1-255
PlayerApplicationSettingAttributeID1	1	Player application setting attribute ID for which the value needs to be set	Valid PlayerApplicationSettingAttributeID values received from the target, or defined as part of Appendix 2
PlayerApplicationSettingValueID1	1	Player application setting value ID for the corresponding player application setting attribute ID	Valid PlayerApplicationSettingValueID values received from the target, or defined as part of Appendix 2

And so on for the number of target defined player application setting attributes and their values.

Response format (SetPlayerApplicationSettingValue)

Parameters	Size(byte)	Description	Allowed Values
None			

NOTE: Setting of a value by CT does not implicitly mean that the setting will take effect on TG. The setting shall take effect after a play command from CT. If currently playing, it is up to the TG to decide when the setting shall take effect. There shall be an error response sent back if there are errors in attribute and/or value.

5.2.5. GetPlayerApplicationSettingAttributeText (PDU ID: 0x15)

Description:

This primitive requests the target device to provide supported player application setting attribute displayable text for the provided PlayerApplicationSettingAttributeIDs.

NOTE: This command is expected to be used only for extended attributes for menu navigation. It is assumed that all <attribute, value> pairs used for menu extensions are statically defined by TG.

Command format (GetPlayerApplicationSettingAttributeText)

Parameters	Size(byte)	Description	Allowed Values
NumPlayerApplicationSettingAttributes (N)	1	Number of player application setting attribute IDs for which corresponding string is needed	1-255
PlayerApplicationSettingAttributeID1	1	Player application setting attribute ID for which the corresponding attribute displayable text is needed	Valid PlayerApplicationSettingAttributeID values received from the target, or defined attributeID as part of Appendix 2

And so on for the number of needed player application setting attribute ID (N)

Response format (GetPlayerApplicationSettingAttributeText)

Parameters	Size(byte)	Description	Allowed Values
NumPlayerApplicationSettingAttributes (N)	1	Number of attributes provided	1-255
PlayerApplicationSettingAttributeID1	1	Specified the player application setting attribute ID for which the displayable text is returned	1-255
CharacterSetID1	2	Specifies the character set ID to be displayed on CT	Use MIBenum defined in IANA character set document (Refer to InformDisplayableCharacterSet)
PlayerApplicationSettingAttributeStringLength1 (n)	1	Length of the player application setting attribute string	1-255
PlayerApplicationSettingAttributeString1	1-n	Specifies the player application setting attribute string in specified character set.	Any string encoded in specified character set

And so on for the number of target defined player application setting attributes in the requested order (N).

5.2.6. GetPlayerApplicationSettingValueText (PDU ID: 0x16)

Description:

This primitive request the target device to provide target supported player application setting value displayable text for the provided player application setting attribute values.

NOTE: This command is expected to be used only for extended attributes for menu navigation. It is assumed that all <attribute, value> pairs used for menu extensions are statically defined by TG.

Command format (Get PlayerApplicationSettingValueText)

Parameters	Size(byte)	Description	Allowed Values
PlayerApplicationSettingAttributeID	1	Player application setting attribute ID	Player application setting attribute ID as defined in Appendix 2 or received from the target
NumPlayerApplicationSettingValue(N)	1	Number of player application setting values for which corresponding string is needed	1-255
PlayerApplicationSettingValueID1	1	Player application setting value ID for which the corresponding value string is needed	Valid ValueID values received from the target, or defined ValueID as part of Appendix 2

And so on for the number of target defined player application setting values in the requested order (N).

Response format (GetPlayerApplicationSettingValueText)

Parameters	Size(byte)	Description	Allowed Values
NumPlayerApplicationSettingValues (N)	1	Number of player application settings value provided	1-255
PlayerApplicationSettingValueID1	1	Player application setting value ID for which the text is returned	1-255
CharacterSetID1	2	Specifies the character set ID to be displayed on CT	Refer to section 5.2.7 for allowed values
PlayerApplicationSettingValueStringLength1 (n)	1	Length of the player application setting value string	1-255
PlayerApplicationSettingValueString1	1-n	Specifies the player application setting value string in specified character set.	Any string encoded in specified character set

And so on for the number of target defined player application setting values in the requested order (N).

5.2.7. InformDisplayableCharacterSet (PDU ID: 0x17)

Description:
This primitive informs the list of character set supported by CT to TG. This shall allow TG to send responses with strings in the character set supported by CT.

When the TG receives this command, the TG can send a string in the character set that is specified in this command. If there is no character set which CT has, the TG will send a string in UTF-8. By default TG shall send strings in UTF-8 if this command has not been sent by CT to TG.

Command format (InformDisplayableCharacterSet)

Parameters	Size(byte)	Description	Allowed Values
NumCharacterSet(N)	1	Number of displayable character sets	1-255
CharacterSetID	2*N	Specifies the character set ID to be displayed on CT.	Use MIBenum defined in IANA character set document. 3: ISO 646 (ASCII) 4: ISO_8859-1 15: JIS_X0201 17: Shift_JIS 36: KS_C_5601-1987 106: UTF-8 1013: UTF-16BE 2025: GB2312 2026: Big5 (in Decimal)

Response format (InformDisplayableCharacterSet)

Parameters	Size(byte)	Description	Allowed Values
None			

Note:
If this command is not issued, UTF-8 shall be used for any strings as default character set. It is mandatory for CT to send UTF-8 as one of the supported character set in the PDU parameters.

The CT should send this command before it sends any commands that support multiple character sets as follows,

- GetPlayerApplicationSettingAttributeText (0x15)
- GetPlayerApplicationSettingValueText (0x16)
- GetElementAttributes (0x20)

CharacterSetID parameter in all the above listed PDUs including this PDU is MIBenum value of the character set defined in IANA character set document [8].

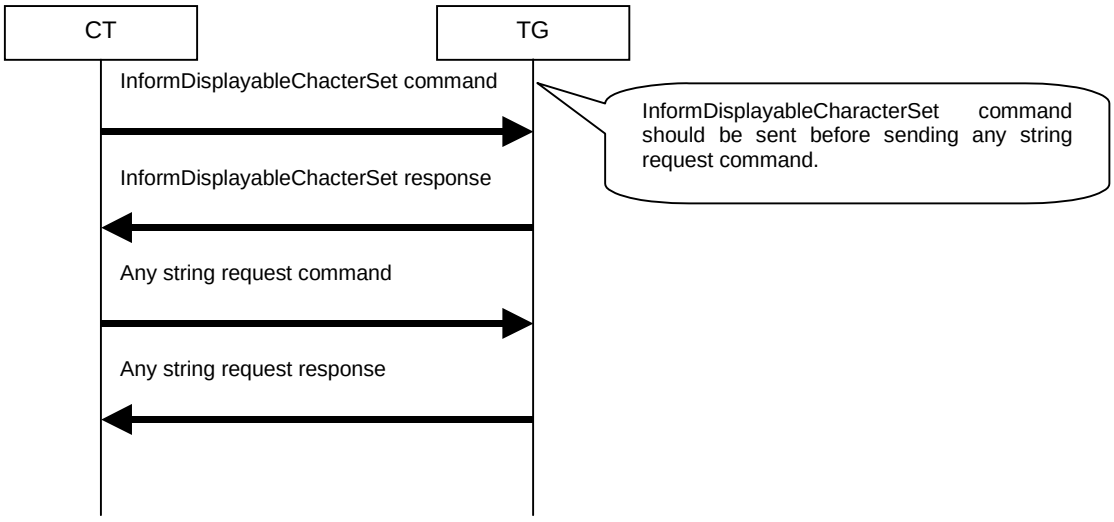

Figure 3 example of using InformDisplayableCharacterSet

5.2.8. InformBatteryStatusOfCT (PDU ID: 0x18)

Description:
This command frame is being sent by the CT to TG whenever the CT's battery status has been changed.

Command format (InformBatteryStatusOfCT)

Parameters	Size(byte)	Description	Allowed Values
Battery status	1	Battery status	0x0 – NORMAL – Battery operation is in normal state 0x1 – WARNING - Unable to operate soon. Specified when battery going down. 0x2 – CRITICAL – Can not operate any more. Specified when battery going down. 0x3 – EXT ERNAL – Connecting to external power supply. 0x4 - FULL_CHARGE – When the device is completely charged.

Response Format (InformBatteryStatusOfCT)

Parameters	Size(byte)	Description	Allowed Values
None			

5.3. Media Information PDUs

The Media Information PDU's are used to obtain detailed information on a particular media file like song information including title, album, artist, composer, year etc.

5.3.1. GetElementAttributes (PDU ID: 0x20)

Description:
These primitive requests the TG to provide the attributes of the element specified in the parameter.

Command format (GetElemenAttributes)

Parameters	Size(byte)	Description	Allowed Values
Identifier	8	Unique identifier to identify an element on TG	PLAYING (0x0): This should return attribute information for the element which is current track in the TG device. All other values other than 0x0 are currently reserved.
NumAttributes (N)	1	Number of Attributes provided	If NumAttributes is set to zero, all attribute information shall be returned, else attribute information for the specified attribute IDs shall be returned by the TG
AttributeID1	4	Specifies the attribute ID for the attributes to be retrieved	See Appendix 1 for list of possible attribute IDs

And so on for each attribute (1…N).

Response Format (GetElementAttributes)

Parameters	Size(byte)	Description	Allowed Values
NumAttributes(N)	1	Number of attributes provided	1-255
AttributeID1	4	Specifies the attribute ID to be written	See Appendix 1 for list of possible attribute IDs
CharacterSetID1	2	Specifies the character set ID to be displayed on CT	Use MIBenum defined in IANA character set document (Refer to InformDisplayableCharacterSet)
AttributeValueLength 1 (n1)	2	Length of the value of the attribute	0-65535 (0, if no name is provided)
AttributeValue1	1-n1	Attribute Name in specified character set	Any text encoded in specified character set
AttributeID2	4	Specifies the attribute ID to be written	See Appendix 1 for list of possible attribute IDs
CharacterSetID2	2	Specifies the character set ID to be displayed on CT	Use MIBenum defined in IANA character set document (Refer to InformDisplayableCharacterSet)
AttributeValueLength 2 (n2)	2	Length of the value of the attribute	0-65535 (0, if no name is provided)
AttributeValue2	1-n2	Attribute value in specified character set.	Any text encoded in specified character set

And so on for all the attributes provided (1…N)

5.4. Notification PDUs

The Notification PDUs are used to obtain asynchronous updates from the TG based on change of status at the target's side. For example, when CT might be interested to know when media track gets changed, so that new media information can be displayed on the controllers display. The CT registers with the TG to receive one or more notifications. The TG sends a notification PDU when a status change happens if the CT had registered for that change.

5.4.1. GetPlayStatus (PDU ID: 0x30)

Description:
This primitive is used by the CT to get the status of the currently playing media at the TG.

Command format (GetPlayStatus)

Parameters	Size(byte)	Description	Allowed Values
None			

Response Format (GetPlayStatus)

Parameters	Size(byte)	Description	Allowed Values
SongLength	4	The total length of the playing song in milliseconds	0-(2^{32} – 1)
SongPosition	4	The current position of the playing in milliseconds elapsed	0-(2^{32} – 1)
PlayStatus	1	Current Status of playing	0x00 : STOPPED 0x01 : PLAYING 0x02 : PAUSED 0x03: FWD_SEEK 0x04: REV_SEEK 0xFF : ERROR

Note:
If TG does not support SongLength And SongPosition on, TG then TG shall return 0xFFFFFFFF.

5.4.2. RegisterNotification (PDU ID: 0x31)

Description:
This primitive registers with the TG to receive notifications asynchronously based on specific events occurring. A registered notification gets changed on receiving CHANGED event notification. This is as per 1394 AV/C protocol specification. Only one EventID shall be used per notification registration. The INTERIM response for this command must be a Notify command with current status or REJECTED response.

Parameters	Size(byte)	Description	Allowed Values
EventID	1	Event for which the CT requires notifications	The event ID are following EVENT_PLAYBACK_STATUS_CHANGED (0x01): Event for change in playback status EVENT_TRACK_CHANGED (0x02): Event for change in track EVENT_TRACK_REACHED_END (0x03): Event for reach to end of the current track. EVENT_TRACK_REACHED_START (0x04): Event for reach to start of the current track. EVENT_PLAYBACK_POS_CHANGED (0x05): Event for change in playback position EVENT_BATT_STATUS_CHANGED (0x06): Event for change in battery status EVENT_SYSTEM_STATUS_CHANGED (0x07): Event for change in system status

			EVENT_PLAYER_APPLICATION_SETTING_CHANGED (0x08): Event for change in player application setting
Playback interval	4	Specifies the time interval (in seconds) at which the change in playback position will be notified. If the song is being forwarded / rewound, a notification will be received whenever the playback position will change by this value. (Applicable only for EventID EVENT_PLAYBACK_POS_CHANGED. For other events , value of this parameter is ignored)	0 < Playback interval Don't use 0.

*Note that TG may send reject response when CT re-register events with RegisterNotification command to TG.

Response Format (RegisterNotification)

Response Data format for EVENT_PLAYBACK_STATUS_CHANGED

Parameters	Size(byte)	Description	Allowed Values
EventID	1	Specific EventID	EVENT_PLAYBACK_STATUS_CHANGED (0x01)
PlayStatus	1	Indicates the current status of playback	0x00: STOPPED 0x01: PLAYING 0x02: PAUSED 0x03: FWD_SEEK 0x04: REV_SEEK 0xFF: ERROR

Response Data format for EVENT_TRACK_CHANGED

Parameters	Size(byte)	Description	Allowed Values
EventID	1	Specific EventID	EVENT_TRACK_CHANGED(0x02)
Identifier	8	Index of the current track	If no track currently selected, then return 0xFFFFFFFF in the INTERIM response.

Response Data format for EVENT_TRACK_REACHED_END

Parameters	Size(byte)	Description	Allowed Values
EventID	1	Specific EventID	EVENT_TRACK_REACHED_END(0x03)
None			

Response Data format for EVENT_TRACK_REACHED_START

Parameters	Size(byte)	Description	Allowed Values
EventID	1	Specific EventID	EVENT_TRACK_REACHED_START (0x04)
None			

Response Data format for EVENT_ PLAYBACK_POS_CHANGED

Parameters	Size(byte)	Description	Allowed Values
EventID	1	Specific EventID	EVENT_PLAYBACK_POS_CHANGED (0x05)
Playback position	4	Current playback position in millisecond	If no track currently selected, then return 0xFFFFFFFF in the INTERIM response.

EVENT_PLAYBACK_POS_CHANGD shall be notified in the following conditions:

- TG has reached the registered playback Interval time.
- Changed PLAY STATUS.
- Changed Current Track.
- Reached end or beginning of track.

Response Data format for EVENT_BATT_STATUS_CHANGED

Parameters	Size(byte)	Description	Allowed Values
EventID	1	Specific EventID	EVENT_BATT_STATUS_CHANGED (0x06)
Battery status	1	Battery status	0x0 – NORMAL – Battery operation is in normal state 0x1 – WARNING - Unable to operate soon. Is provided when the battery level is going down. 0x2 – CRITICAL – Can not operate any more. Is provided when the battery level is going down. 0x3 – EXT ERNAL – Connecting to external power supply 0x4 - FULL_CHARGE – When the device is completely charged from the external power supply.

NOTE: Battery status notification defined in this specification is expected to be replaced by Attribute profile specification in the future.

Response Data format for EVENT_SYSTEM_STATUS_CHANGED

Parameters	Size(byte)	Description	Allowed Values
EventID	1	Specific EventID	EVENT_SYSTEM_STATUS_CHANGED (0x07)
SystemStatus	1	Indicates the current System status.	POWER_ON (0x00) POWER_OFF (0x01) UNPLUGGED (0x02)

POWER_OFF and UNPLUGGED are used for Bluetooth Accessories which attach to Media Players. In this case, it will happen that Audio Player's power state is "POWER OFF" or Audio Player is detached from Bluetooth Adapter (UNPLUGGED)

Response Data format for EVENT_PLAYER_APPLICATION_SETTING_CHANGED

Parameters	Size(byte)	Description	Allowed Values
EventID	1	Specific EventID	EVENT_PLAYER_APPLICATION_SETTING_CHANGED (0x08)
NumPlayerApplicationSettingAttributes(N)	1	Number of player application setting attributes that follow	1-255
PlayerApplicationSettingAttributeID1	1	Player application setting attribute ID for which the value is returned	1-255
PlayerApplicationSettingValueID1	1	Currently set player application setting value on the target for the corresponding requested PlayerApplicationSettingAttribute ID	Valid PlayerApplicationSettingValueID values, or defined as part of Appendix 2

And so on for the number of target defined player application setting values.

5.5. Continuing PDUs

5.5.1. RequestContinuingResponse (PDU ID: 0x40)

Description:
This primitive is used by CT to request continuing response that has not completed. If it requested, a series of part is responded by operational command. This command will be invoked by CT after receiving a response with <Packet Type – First (0x01) or Continue (0x10)>.

Command format (RequestContinuingResponse)

Parameters	Size(byte)	Description	Allowed Values
ContinuePDU_ID	1	Target PDU_ID for continue command	PDU_ID

Response Format (RequestContinuingResponse)

The response for this command is the pending data for the previous command invoked by CT. (Refer to Figure 4)

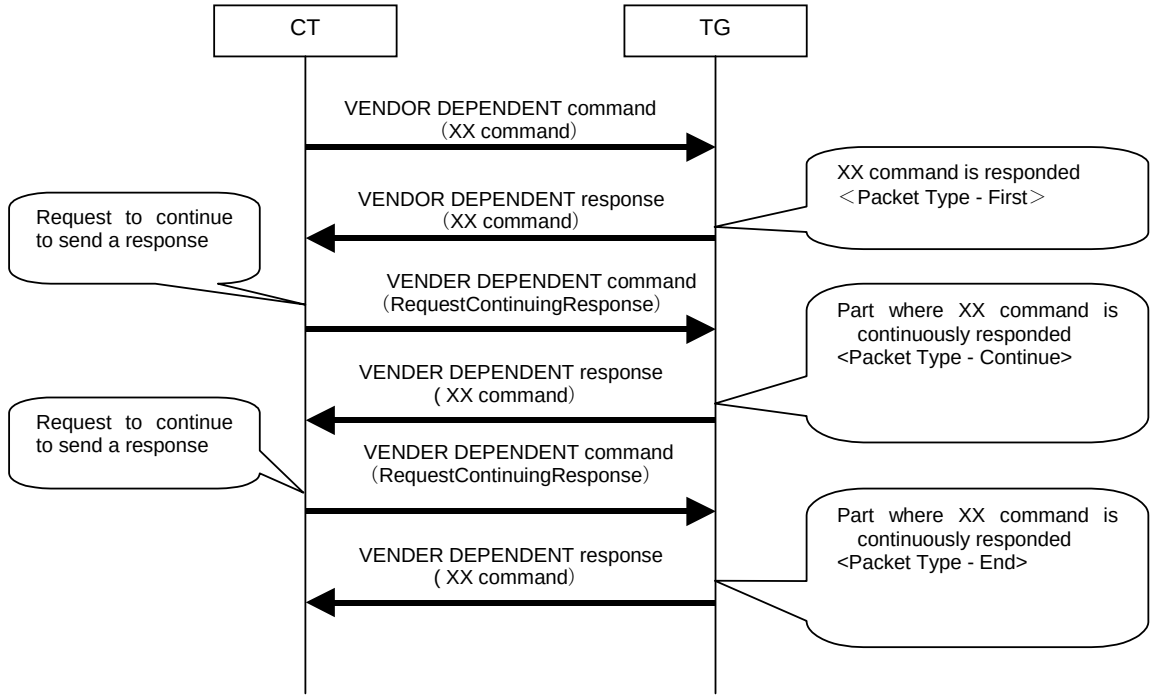

Figure 4 Request for sending a response continuously

5.5.2. AbortContinuingResponse (PDU ID: 0x41)

Description:
This primitive is used by CT to abort continuing response. This command will be invoked by CT after receiving a response with <Packet Type – First (0x01) or Continue (0x10)>.
(Refer to Figure 5)

Command format (AbortContinuingResponse)

Parameters	Size(byte)	Description	Allowed Values
ContinueAbort PDU_ID	1	Target PDU_ID for abort continue command	PDU_ID

Response Format (AbortContinuingResponse)

Parameters	Size(byte)	Description	Allowed Values
None			

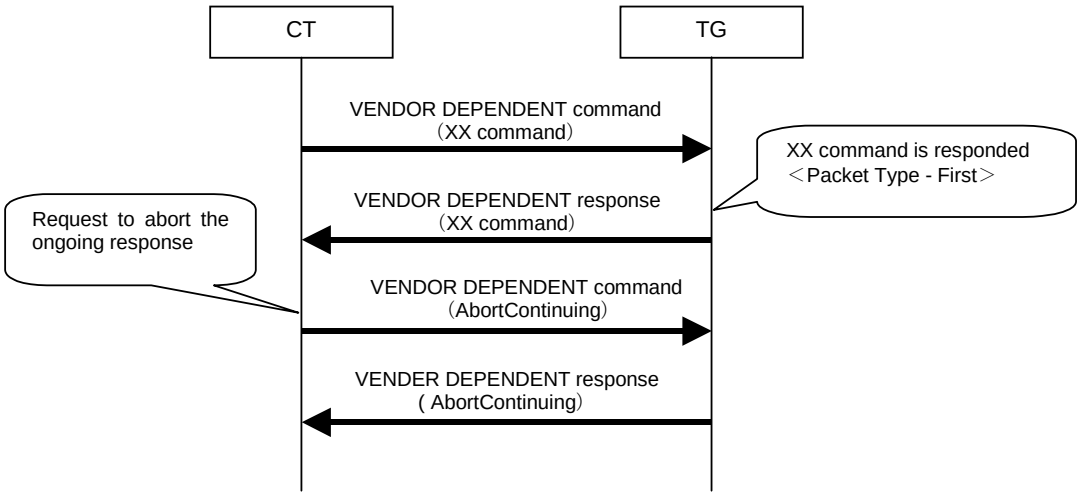

Figure 5 AbortContinuingResponse

5.6. Basic Group Navigation

Basic group navigation PDUs are defined to support a logical one dimensional group structure of media content on the TG to CT for easier navigation purpose. The definition of groups on the TG is implementation dependent. The group structure can consist of parts of, or a mix of playlists and artist/album/genre folders etc that are used by the media player applications in the TG.

The basic group navigation PDUs have a similar behavior as the Forward and Backward commands, but instead of navigating to the next/previous song they are used to navigate to the first song in the next/previous group.

The Basic Group Navigation PDUs are transported as vendor unique PASS THROUGH commands.

5.6.1. Next Group (vendor unique id: 0x00)

Description:
This function is used to move to the first song in the next group. The definition of group on TG is implementation dependent.

5.6.2. Previous Group (vendor unique id: 0x01)

Description:
This function is used to move to the first song in the previous group. The definition of group on TG is implementation dependent.

6. Error handling

- If CT sent a PDU with nonexistent PDU ID or a PDU containing only one parameter with nonexistent parameter ID, TG shall return REJECTED response with Error Status Code. TG may return REJECTED response also in other situations (See 7. Error Status Code).
- If CT sent a PDU with multiple parameters where at least one ID is existent and the others are nonexistent, TG shall proceed with the existent ID and ignore the non-existent IDs.

Note, that CT can always have complete information which IDs were accepted by TG: in case of STATUS PDUs the response will contain information for the IDs which were understood, when setting values for Player application settings; TG will return notification response with the list of AttributeIDs for which values have been set.

7. Error Status Code

7.1. Error Status Code

An error status code is added to the REJECTED response if TG rejected the command. It is useful for CT to know why the command is rejected by TG. Table 7.1.1 below shows the error status code.

Error Code Value	Description
0x00	Invalid command, sent if TG received a PDU not it did not understand.
0x01	Invalid parameter, sent if the TG received a PDU with a parameter ID that it did not understand. Sent if there is only one parameter ID in the PDU.
0x02	Specified parameter not found, sent if the parameter ID is understood, but content is wrong or corrupted.
0x03	Internal Error, sent if there are other error conditions.

Table 7.1.1 List of Error Status Code

An example of response packet format for REJECTED will be as below.

Oct	MSB (7)	6	5	4	3	2	1	LSB (0)
0	0x0				Response: 0xA(REJECTED)			
1	Subunit_type:0x9 (PANEL)					Subunit_ID: 0x0		
2	Opcode: 0x0 (VENDOR DEPENDENT)							
3 – 5	Company ID: 0x0017A7 [MCPC registered CompanyID]							
6	PDU ID (of the command for which this response is sent)							
7	Reserved (0x00)						Packet Type (0x0)	
8 – 9	Parameter Length (0x0001)							
10	Error Code (0x02 – Specified parameter not found)							

Table 7.1.2 An example of response packet format for REJECTED

8. List of Figures

9. List of Tables

10. Appendix 1

10.1. Attribute IDs

The table below provides the list of IDs for Attributes. These IDs are used to uniquely identify media information.

Attribute ID	Description	Allowed values	Mandatory / Optional
0x0	Illegal , Should not be used	-	
0x1	Title of the media	Any text encoded in specified character set	M
0x2	Name of the artist	Any text encoded in specified character set	O
0x3	Name of the album	Any text encoded in specified character set	O
0x4	Number of the media (ex. Track number of the CD)	Numeric ASCII text with zero suppresses	O
0x5	Total number of the media (ex. Total track number of the CD)	Numeric ASCII text with zero suppresses	O
0x6	Genre	Any text encoded in specified character set	O
0x7	Playing time in millisecond	Numeric ASCII text (Ex. 2min30sec → 150000)	O
0x8-0xFFFFFFFF	Reserved for future use	-	

Table 10.1.1 Attribute IDs

NOTE: If the track title is not available the TG shall try to identify the track in other ways or send information about the media. If no information is available an empty string of zero length may be sent.

11. Appendix 2

11.1. PlayerApplicationSettingAttributeIDs

The table below provides the list of IDs for device settings. These IDs are used to uniquely identify and exchange information on player application settings between the TG and the CT.

PlayerApplicationSetting Attribute	Attribute Description	Defined Values
0x00	Illegal , Should not be used	None
0x01	Equalizer ON/OFF status	PlayerApplicationSettingValueID ValueID: Description 0x01: OFF 0x02: ON 0x03-0xFF: Reserved for future use
0x02	Repeat Mode status	PlayerApplicationSettingValueID ValueID: Description 0x01: OFF 0x02: Single track repeat 0x03: All track repeat 0x04: Group repeat 0x05-0xFF: Reserved for future use
0x03	Shuffle ON/OFF status	PlayerApplicationSettingValueID ValueID: Description 0x01: OFF 0x02: All tracks shuffle 0x03: Group shuffle 0x04-0xFF: Reserved for future use
0x04	Scan ON/OFF status	PlayerApplicationSettingValueID ValueID: Description 0x01: OFF 0x02: All tracks scan 0x03: Group scan 0x04-0xFF: Reserved for future use
0x05 – 0x7F	Reserved for future use	
0x80 – 0xFF	Vendor specified	

Table 11.1.1 PlayerApplicationSettingAttributeIDs

12. Appendix 3

12.1. Message Sequence Chart (MSC)

Below is an example MSC how to follow to track information on CT in case track change is occurred by PASSTHROUGH command.

(NOTE) THIS MSC IS JUST A ONE OF EXAMPLES. ALL BEHAVIORS BOTH OF CT AND TG are IMPLEMENTATION DEPENDENT.

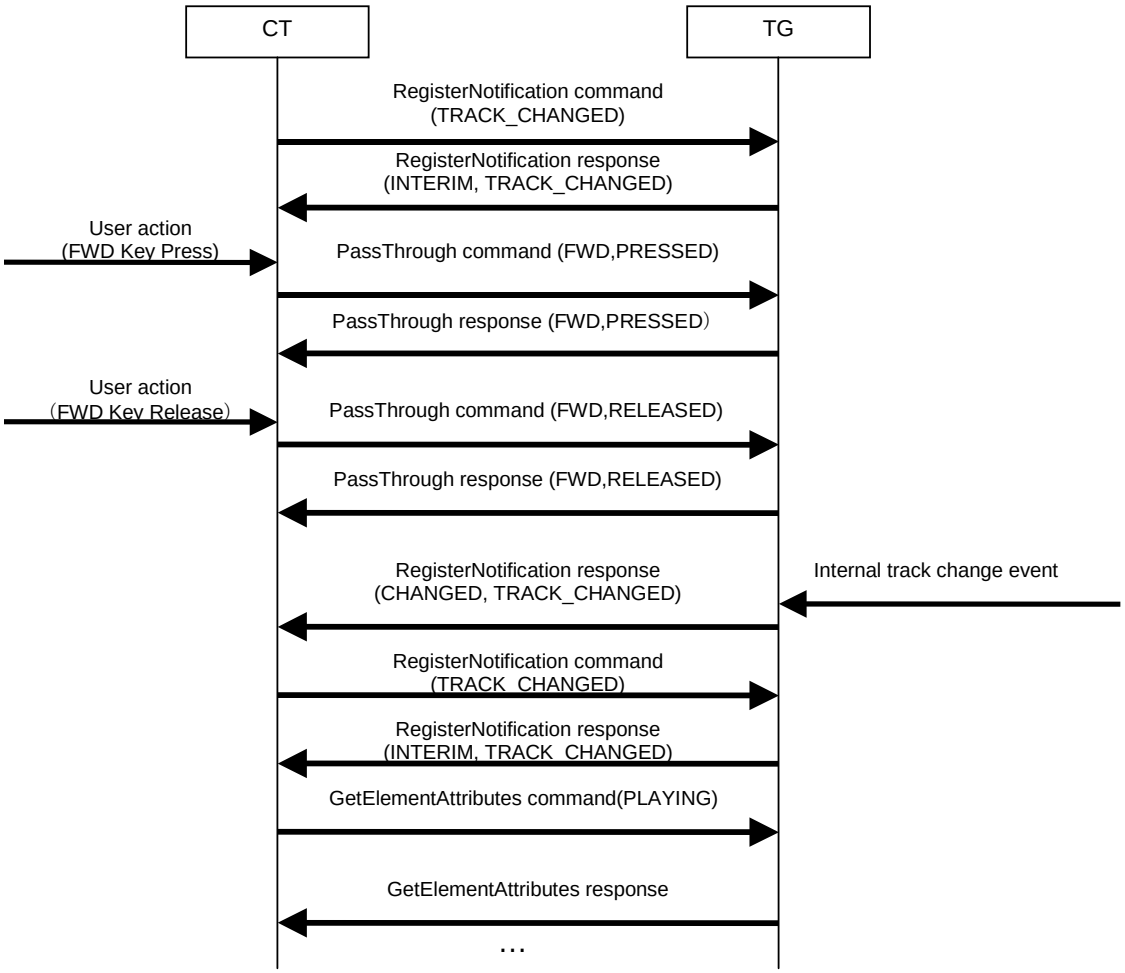

Figure 6 Example Message Sequence Chart

12.2. Player Application Setting PDUs

Below is an example MSC how to manipulate player application settings using Player Application Setting PDUs.

The table below is an example player application setting of TG.

PlayerApplicationSettingAttributeID	**PlayerApplicationSettingAttributeText**	**PlayerApplicationSettingValueID**	
0x01	“Equalizer”	**PlayerApplicationSettingValueID**	**PlayerApplicationSettingValueText**
		0x01	“OFF”
		0x02	“ON”
0x03	“Shuffle Mode”	**PlayerApplicationSettingValueID**	**PlayerApplicationSettingValueText**
		0x01	“OFF”
		0x02	“All tracks shuffle”
0x80	"Language Setting"	**PlayerApplicationSettingValueID**	**PlayerApplicationSettingValueText**
		0x01	"eng"
		0x02	"jpn"
		0x03	"fra"
0x81	“Sleep Timer”	**PlayerApplicationSettingValueID**	**PlayerApplicationSettingValueText**
		0x01	“OFF”
		0x02	“20 min.”

Table 12.2.1 example device setting of TG

MSC are described as follows.

<table>
<tr><th>CT</th><th>TG</th></tr>
<tr><td></td><td></td></tr>
<tr><td colspan="2">[Get a player application setting list and retrieve text labels for each.]</td></tr>
<tr><td>GetListPlayerApplicationSettingAttributes(Command)<table><tr><td>None</td><td></td></tr></table></td><td></td></tr>
<tr><td></td><td>GetListPlayerApplicationSettingAttributes(Response)<table><tr><td>NumPlayerApplicationSettingAttributes</td><td>0x04</td></tr><tr><td>PlayerApplicationSettingAttributeID1</td><td>0x01</td></tr><tr><td>PlayerApplicationSettingAttributeID2</td><td>0x03</td></tr><tr><td>PlayerApplicationSettingAttributeID3</td><td>0x80</td></tr><tr><td>PlayerApplicationSettingAttributeID4</td><td>0x81</td></tr></table></td></tr>
<tr><td>GetPlayerApplicationSettingAttributeText (Command)<table><tr><td>NumPlayerApplicationSettingAttribute</td><td>0x04</td></tr><tr><td>PlayerApplicationSettingAttributeID1</td><td>0x01</td></tr><tr><td>PlayerApplicationSettingAttributeID2</td><td>0x03</td></tr><tr><td>PlayerApplicationSettingAttributeID3</td><td>0x80</td></tr><tr><td>PlayerApplicationSettingAttributeID4</td><td>0x81</td></tr></table></td><td></td></tr>
<tr><td></td><td>GetPlayerApplicationSettingAttributeText (Response)<table><tr><td>NumPlayerApplicationSettingAttributes</td><td>0x04</td></tr><tr><td>PlayerApplicationSettingAttributeID1</td><td>0x01</td></tr><tr><td>CharacterSetID1</td><td>0x006a (UTF-8)</td></tr><tr><td>PlayerApplicationSettingAttributeStringLength1</td><td>0x09</td></tr><tr><td>PlayerApplicationSettingAttributeString1</td><td>“Equalizer”</td></tr><tr><td>PlayerApplicationSettingAttributeID2</td><td>0x03</td></tr><tr><td>CharacterSetID2</td><td>0x006a (UTF-8)</td></tr><tr><td>PlayerApplicationSettingAttributeStringLength2</td><td>0x0c</td></tr><tr><td>PlayerApplicationSettingAttributeString2</td><td>“Shuffle Mode”</td></tr><tr><td>PlayerApplicationSettingAttributeID3</td><td>0x80</td></tr><tr><td>CharacterSetID3</td><td>0x006a (UTF-8)</td></tr><tr><td>PlayerApplicationSettingAttributeStringLength3</td><td>0x10</td></tr><tr><td>PlayerApplicationSettingAttributeString3</td><td>"Language Setting"</td></tr></table></td></tr>
</table>

	PlayerApplicationSettingAttributeID4: 0x81 CharacterSetID4: 0x006a (UTF-8) PlayerApplicationSettingAttributeStringLength4: 0x0b PlayerApplicationSettingAttributeString4: "Sleep Timer"
[Get a value of all settings and get text labels of each value.]	
ListPlayerApplicationSettingValues (Command) PlayerApplicationSettingAttributeID: 0x01 (Equalizer)	
	ListPlayerApplicationSettingValues (Response) NumPlayerApplicationSettingValues: 0x02 PlayerApplicationSettingValueID1: 0x01 PlayerApplicationSettingValueID2: 0x02
ListPlayerApplicationSettingValues (Command) PlayerApplicationSettingAttributeID: 0x03 (Shuffle Mode)	
	ListPlayerApplicationSettingValues (Response) NumPlayerApplicationSettingValues: 0x02 PlayerApplicationSettingValueID1: 0x01 PlayerApplicationSettingValueID2: 0x02
ListPlayerApplicationSettingValues (Command) PlayerApplicationSettingAttributeID: 0x80 (Language Setting)	
	ListPlayerApplicationSettingValues (Response) NumPlayerApplicationSettingValues: 0x03 PlayerApplicationSettingValueID1: 0x01 PlayerApplicationSettingValueID2: 0x02 PlayerApplicationSettingValueID3: 0x03
ListPlayerApplicationSettingValues (Command) PlayerApplicationSettingAttributeID: 0x81 (Sleep Timer)	
	ListPlayerApplicationSettingValues (Response) NumPlayerApplicationSettingValues: 0x02 PlayerApplicationSettingValueID1: 0x01 PlayerApplicationSettingValueID2: 0x02
GetPlayerApplicationSettingValueText	

(Command)

PlayerApplicationSettingAttributeID	0x01 (Equalizer)
NumPlayerApplicationSettingValue	0x02
PlayerApplicationSettingValueID1	0x01
PlayerApplicationSettingValueID2	0x02

GetPlayerApplicationSettingValueText (Response)

NumPlayerApplicationSettingValues	0x02
PlayerApplicationSettingValueID1	0x01
CharacterSetID1	0x006a (UTF-8)
PlayerApplicationSettingValueStringLength1	0x03
PlayerApplicationSettingValueString1	“OFF”
PlayerApplicationSettingValueID2	0x02
CharacterSetID2	0x006a (UTF-8)
PlayerApplicationSettingValueStringLength2	0x02
PlayerApplicationSettingValueString2	“ON”

(similar sequence will continue to retrieve all PlayerApplicationSettingValueString)

[Get current device setting values.]

GetCurrentPlayerApplicationSettingValue (Command)

NumPlayerApplicationSettingAttributeID	0x04
PlayerApplicationSettingAttributeID1	0x01
PlayerApplicationSettingAttributeID2	0x03
PlayerApplicationSettingAttributeID3	0x80
PlayerApplicationSettingAttributeID4	0x81

GetCurrentPlayerApplicationSettingValue (Response)

NumPlayerApplicationSettingValues	0x04
PlayerApplicationSettingAttributeID1	0x01 (Equalizer)
PlayerApplicationSettingValueID1	0x02 (ON)
PlayerApplicationSettingAttributeID2	0x03 (Shuffle Mode)
PlayerApplicationSettingValueID2	0x01 (OFF)
PlayerApplicationSettingAttributeID3	0x80 (Language Setting)
PlayerApplicationSettingValueID3	0x01 (eng)
PlayerApplicationSettingAttributeID4	0x81 (Sleep Timer)
PlayerApplicationSettingValueID4	0x01 (OFF)

[Set device setting values.]

SetPlayerApplicationSettingValue (Command)

NumPlayerApplicationSettingAttributeID	0x04
PlayerApplicationSettingAttributeID1	0x01 (Equalizer)
PlayerApplicationSettingValueID1	0x02 (ON)
PlayerApplicationSettingAttributeID2	0x03 (Shuffle Mode)
PlayerApplicationSettingValueID2	0x02 (All track shuffle)
PlayerApplicationSettingAttributeID3	0x80 (Language Setting)
PlayerApplicationSettingValueID3	0x02 (jpn)
PlayerApplicationSettingAttributeID4	0x81 (Sleep Timer)
PlayerApplicationSettingValueID4	0x02 (20 min.)

SetPlayerApplicationSettingValue (Response)

None	

Table 12.2.2 example MSC to handle device setting using PlayerApplicationSettings PDUs

Register Notification to follow the device setting of TG.

<table>
<tr><th>CT</th><th>TG</th></tr>
<tr><td>RegisterNotificaion (Command)<table><tr><td>EventID</td><td>0x08(EVENT_DEVICE_SETTING_CHANGED</td></tr><tr><td>Playback interval</td><td>Any (this value is ignored)</td></tr></table></td><td></td></tr>
<tr><td></td><td>RegisterNotificaion (Response) - INTERIM<table><tr><td>EventID</td><td>0x08</td></tr><tr><td>NumDeviceAttribute</td><td>0x04</td></tr><tr><td>PlayerApplicationSettingAttributeID1</td><td>0x01 (Equalizer)</td></tr><tr><td>PlayerApplicationSettingValueID1</td><td>0x01 (OFF)</td></tr><tr><td>PlayerApplicationSettingAttributeID2</td><td>0x03 (Shuffle Mode)</td></tr><tr><td>PlayerApplicationSettingValueID2</td><td>0x01 (OFF)</td></tr><tr><td>PlayerApplicationSettingAttributeID3</td><td>0x80 (Language Setting)</td></tr><tr><td>PlayerApplicationSettingValueID3</td><td>0x01 (eng)</td></tr><tr><td>PlayerApplicationSettingAttributeID4</td><td>0x81 (Sleep Timer)</td></tr><tr><td>PlayerApplicationSettingValueID4</td><td>0x01 (OFF)</td></tr></table></td></tr>
<tr><td colspan="2">[Player Application settings are changed by user action.]
Equalizer “OFF” → “ON”</td></tr>
<tr><td></td><td>RegisterNotificaion (Response) - CHANGED<table><tr><td>EventID</td><td>0x08</td></tr><tr><td>NumPlayerApplicationSettingAttribute</td><td>0x04</td></tr><tr><td>PlayerApplicationSettingAttributeID1</td><td>0x01 (Equalizer)</td></tr><tr><td>PlayerApplicationSettingValueID1</td><td>0x02 (ON)</td></tr><tr><td>PlayerApplicationSettingAttributeID2</td><td>0x03 (Shuffle Mode)</td></tr><tr><td>PlayerApplicationSettingValueID2</td><td>0x01 (OFF)</td></tr><tr><td>PlayerApplicationSettingAttributeID3</td><td>0x80 (Language Setting)</td></tr><tr><td>PlayerApplicationSettingValueID3</td><td>0x01 (eng)</td></tr><tr><td>PlayerApplicationSettingAttributeID4</td><td>0x81 (Sleep Timer)</td></tr><tr><td>PlayerApplicationSettingValueID4</td><td>0x01 (OFF)</td></tr></table></td></tr>
</table>

Table 12.2.3 example MSC to follow the device setting of TG

13. Appendix 4

The table below gives the list of EventIDs defined in this specification to be supported by TG.

EventID	Description	Mandatory / Optional
EVENT_PLAYBACK_STATUS_CHANGED(0x01)	Change in playback status of the current track.	M
EVENT_TRACK_CHANGED (0x02)	Change of current track	M
EVENT_TRACK_REACHED_END (0x03)	Reached end of a track	O
EVENT_TRACK_REACHED_START (0x04)	Reached start of a track	O
EVENT_PLAYBACK_POS_CHANGED (0x05)	Change in playback position. Returned after the specified playback notification change notification interval	O
EVENT_BATT_STATUS_CHANGED (0x06)	Change in battery status	O
EVENT_SYSTEM_STATUS_CHANGED (0x07)	Change in system status	O
EVENT_PLAYER_APPLICATION__SETTING_CHANGED(0x08)	Change in player application setting	O
0x09-0xFF	Reserved for future use	

14. Appendix 5

14.1. AV/C Transaction Rules

An AV/C transaction consists of one message containing a command frame addressed to the TG and zero or more messages containing a response frame returned to the CT by the TG. The TG is required to generate a response frame within specified time periods

All transactions except VENDOR DEPENDENT commands shall comply with the following time period. TG shall respond to any command within a time period of T_{RCP} (100) counting from the moment a command frame is received.

For some transactions, the TG may not be able to complete the request or determine whether it is possible to complete the request within the T_{RCP} (100) allowed. In this case, the TG shall return an initial response code in INTERIM with the expectation that the final response follow later.

14.2. Timers and Counters

The following timer is required by AVRCP.

Timer Name	Proposed Value	Description
T_{RCP} (100)	100 milliseconds	A TG shall return its response frame within 100 milliseconds counting from the receipt of the command frame.
T_{MTC} (200)	200 milliseconds	A TG shall return its response frame within 200 milliseconds counting from the receipt of the command frame.
T_{MTP} (1000)	1000 milliseconds	A TG shall return its response frame within 1000 milliseconds counting from the receipt of the command frame.

There are no AVRCP specific counters.

15. References

[1] Bluetooth SIG, Specification of the Bluetooth System, Core, Version 1.0, Audio/Video Control Transport Protocol (http://www.bluetooth.org)
[2] Bluetooth SIG, Specification of the Bluetooth System, Core, Version 1.0, Audio/Video Remote Control Profile (http://www.bluetooth.org)
[3] 1394 Trade Association, AV/C Digital Interface Command Set - General Specification, Version 4.0, Document No. 1999026 (http://www.1394ta.org)
[4] 1394 Trade Association, AV/C Panel Subunit, Version 1.1, Document No. 2001001 (http://www.1394ta.org)
[5] IANA (Internet Assigned Numbers Authority) CHARACTER SETS (http://www.iana.org/assignments/character-sets)
[6] ISO Standard ISO/FDIS 639-2 from the technical committee/subcommittee TC 37 / SC 2 (http://www.id3.org/iso639-2.html)
[7] ID3 tag version 2.4.0 (http://www.id3.org/id3v2.4.0-frames.txt)
[8] http://www.iana.org/assignments/character-sets