RIO Protocol

For Third Party Integrators

Revision 1.16.01

Russound | 1 Forbes Road Newmarket, NH 03857 | 603.659.5170 |

Document History

Note: Version identifiers are used throughout the document to indicate when a feature was introduced.

Revision	Date	Author	Version	Description
1.00.00	12/11/09	bille	v1.0	Initial Release
1.01.00	7/6/10	bille	v1.1	Added support for Power Management
1.02.00	6/23/11	bille	v1.2	Added ability to GET most controller, zone, source and system parameters. Added DMS-3.1 support. Added clarification for case-sensitivity. Corrected case in S[s].type=”Home Control” to ”Home Control” Added clarification for using Media Management (one session per connection) Updated definitions of the S[s].channel and S[s].channelName keys Updated use of S[s].composerName
1.03.00	6/30/11	bille	v1.3	Updated DMS-3.1 MM support. Added KeyCode EVENT. Added GET S[s].ipAddress.
1.04.00	10/11/11	bille	v1.4	Added Shuffle support Added MMClose command Added zone page support
1.05.00	4/1/12	bille	v1.5	Added Repeat support (see the ‘EVENT’ section) Added Controller type support (see the ‘GET’ section) Added SET/GET System.language Support

1.06.00	7/15/12	bille	v1.6	Added System.language notification support Added Media Management Long List support Added Media Management Screen Templates Removed ‘STANDBY’ response from GET System.status command
1.07.00	12/10/12 1/9/13	Pfcass	V1.7	Add SleepTimeRemaining and associated Notification Add Table of Numeric Key Codes at end of doc Added Banks and Presets support Added Favorites support Added Zone and Source Exclusion support Added Forms support See: MMUseForms MMFormEntry MMHelpText2 MMForm<n>Field Added Media Rating support See: MMRate MMContextMenu MMButtons Added Hidden Attribute support

1.07.01	7/1/13	Bill R	V1.7	Changed references to MMNextItems and MMPrevItems to KeyRelease PageUp and KeyRelease PageDown for MM Info Screen Removed reference to MMStartChar. This will be available in the RIO 1.9 specification Updated TOC.
1.07.02	3/31/14	Bill R	V1.7	Added controller and zone indices clarification for X5. Added clarification of non-support of RS232 for X5.
1.08.00	6/3/14	Bill R	V1.8	Added System Favorite renaming support via SET system.favorite[<favorite num>].name=”<name>” Increased favorite name allowable size from 12 characters to 50 characters.
1.08.01	10/20/14	Bill R	V1.8	Corrected repeatMode responses. - Original text indicated valid responses were OFF, SINGLE, ON. - Corrected responses to read OFF, SINGLE, ALL. Corrected notation for shuffleMode. - Notation now includes all media streamers instead of just DMS-3.1.
1.08.02	12/26/14	Bill R	V1.8	Added DVR List and CC keycodes Corrected descriptions for Context Menu and Likes/Dislikes. Favorites and Presets: - Moved Managing Favorites and Managing Presets after events section. - Moved Favorite and Preset events to events table. - Added SET System.favorite[f].name to System SET Key Table.

1.09.00	4/20/16	Greg L	V1.9	Added ZoneMuteOff and ZoneMuteOn commands.
1.10.00	8/12/16	Ed H	V1.10	Updated document to reference current product line. Removed legacy source types. Fixed syntax for C[c].type to accurately reflect the model type. See Controller GET Key Table for details.
1.10.01	8/25/16	Ed H	V1.10	Removed MMBtnCancel since it was not implemented. Added Summary of Media Management INFO Screen Notifications
1.10.02	2/13/17	Ed H	V1.10	Media Service Modes and Key Support table omitted artistName key for TuneIn and vTuner. Table was corrected to reflect support of that key for those services.
1.11.00	7/7/17	Ed H	V1.11	Added sections for MMSelectOption and MMKeepAlive Added system favorites to WATCH SYSTEM
1.12.00	8/14/17	Ed H	V1.12	Added System.Support to determine capability for managing device grouping and system alarm. Added support for device grouping. See WATCH Grouping. Added support for alarm.
1.12.01	11/10/17	Ed H, John M	V1.12	Added alarm notification to zone watch.
1.12.02	6/12/18	Ed H	V1.12	Removed alarm notification from zone watch Added isGroupable to deviceList data Added firmwareVersion

1.14.00	3/25/19	Bill R	V1.14	Added new display item attributes for MBX series products. Added playStatus GET and notification descriptions Added availableControls GET and notification descriptions Added sampleRate GET and notification descriptions Added bitRate GET and notification descriptions Added bitDepth GET and notification descriptions Added playTime GET and notification descriptions Added trackTime GET and notification descriptions Added SetSeekTime EVENT description Added menu item context menu via MMSelectItem Updated JSON menu items to include value, imgUrl, uri, and attributes fields. Included new example of JSON menu list with additional fields.
1.14.01	5/10/19	Ed H	V1.14	Added sleepTimeDefault to Zone GET and SET Key Tables Added sleepTime to System.Support.
1.14.02	6/20/19	Bill R	V1.14	Clarified “Select Menu Option” section.
1.14.03	7/31/19	Ed H	V1.14	Removed GroupVolumeUp and GroupVolumeDown from KeyPress EVENT table

1.15.00	10/10/19	Bill R	V1.15	Added Reboot event Added feature to allow for more System Favorite information to be displayed. Added feature to allow retrieval of player data and mechanism to re-apply data as an off-line favorite. Added GET System.favorite[f].source Added “source” field to favorite notification Added GET System.favorite[f].sourceType Added “sourceType” field to favorite notification Added GET System.favorite[f].providerMode Added “providerMode” field to favorite notification Added GET System.favorite[f].albumCoverURL Added “albumCoverURL” field to favorite notification Added GET System.favorite[f].playerData Added “playerData” field to favorite notification Added GET S[s].playerData Added “playerData” to source notification Added “RestorePlayerData” event

1.15.01	10/18/19 10/25/19	Bill R	V1.15	Added GET S[s].Support.MM.restoreOption Added N S[s].Support.restoreOption Added GET S[s].Support.availableControls Added N S[s].Support.availableControls Added GET S[s].Support.trackTime Added N S[s].Support.trackTime Added GET S[s].Support.playTime Added N S[s].Support.playTime Added GET S[s].Support.playerData Added N S[s].Support.playerData Added GET S[s].Support.favoritesV2 Added N S[s].Support.favoritesV2 Several corrections, clarifications and general clean-up
1.15.02	11/21/19	Bill R	V1.15	Added user login attribute
1.15.03	12/11/19	Bill R	V1.15	Added media streamer mode metadata table.
1.15.04	1/2/20	Bill R	V1.15	Added recommended metadata field usage to media streamer mode metadata table..
1.16.00	5/26/20	Ed H.	V1.16	Removed sleepTimeDefault and sleepTimeRemaining from Zone SET Key Table. Added Sleep Timer section.
1.16.01	8/27/20	Ed H.	V1.16	Added mcaZoneList to WATCH GROUPING Added commands for support of Zone Alarms

Table of Contents

Table of Contents....9
Introduction....12
Conventions....13
Command Overview....14
RIO Protocol Syntax....14
RIO Command Response Syntax....15
The 'VERSION' Command....16
The 'GET' Command....17
Controller GET Key Table....18
System GET Key Table....19
Zone Get Key Table....21
Source GET Key Table....23
Source GET Key Table (Banks and Presets)....31
The 'SET' Command....33
System SET Key Table....34
Zone SET Key Table....34
Source SET Key Table....35
The 'ADJUST' Command....37
The 'EVENT' Command....40
Physical vs Logical Source Selection....40
Sleep Timer....41
EVENT Table....42
Key Events....45
AM/FM Tuner Specific Key Events....49
Managing Favorites....50
Managing Banks and Presets....54
The 'WATCH' Command....56
WATCH Notification Messages....59
WATCH System....60
WATCH a Zone....61
WATCH a Source....62

Media Streamer Mode Metadata Support Table 64
WATCH GROUPING 70
Zone Alarms 73
Alarm Status 73
Alarm Configuration 73
Alarm Options 74
Alarm GET Commands 75
Alarm EVENT Commands 75
saveAlarm 75
snoozeAlarm 75
dismissAlarm 76
Alarm Status WATCH 76
Media Management 77
RIO MM Commands and Responses 78
RIO MM Commands 78
Specify Menu Notification Verbosity 78
Specify Menu Indexing Method 79
Specify Menu Notification Max Items 79
Specify Menu Notification Format 79
Specify Info Screen Format 81
Specify Text Entry Format 81
Navigate to a Menu Item by Index 82
MM Session Initialization 82
MM Session Termination 82
Menu Item Selection 83
Setting Optional Value 84
Item Context Menu Selection 84
Screen Data Confirmation 85
Item Navigation 85
Screen Navigation 85
Text Edit Navigation 85
The Context Menu 86

Likes and Dislikes ... 86
Text Field Entry ... 87
Select Menu Option ... 87
Keep Alive ... 88
RIO MM Responses ... 88
Screen Change Notifications ... 88
Screen Title Notifications ... 89
Menu Item Notifications ... 89
Info Screen Notifications ... 95
Now Playing Notifications ... 96
Text Entry Screen Notifications ... 96
Russound Media Streamer Example Screens ... 99
Menu Screen ... 99
MM Info Screen ... 101
Now Playing Screen ... 103
Text Entry Screen ... 105
Summary of Events ... 107
Summary of Media Management Events ... 108
Summary of Media Management INFO Screen Notifications ... 109
RIO Key String ... 114
Using PuTTY as a RIO Client ... 115
Table of Numeric Key Codes ... 116

Introduction

This document provides a description of the protocol recommended for integration of third-party devices and software with Russound MCA-Series, XStream Series and MBX Series products. We refer to this feature as **RIO** (Russound I/O).

RIO is a text-based command set that is used for third party integration support, making it easy to develop user interfaces and controls for select Russound audio products. Here is a summary of the services provided:

- Full 2-way communication capability.
- The RIO Command Set is available as ASCII text via TCP/IP and RS232 interfaces. (RS232 only supported on MCA-Series)
- Up to 64 simultaneous IP connections are supported.
- Adjustable RS232 baud rates. Choose between 19200, 38400, 57600 and 115200.
- TCP/IP port 9621 (XZone4 uses TCP/IP ports 9621-9624).
- Commands for real-time configuration and monitoring of system parameters such as Volume, Zone On/Off, Source Selection, Party Mode, DND, Source/Zone Names, Preamp Controls and more.
- Support for asynchronous notification of activity on a per-zone, per-source and per-system basis.
- Support for Media Management ('Content Browsing and Selection'). This provides a compressed command set, making it easy to develop a UI device capable of mimicking the ***MyRussound™*** app (iOS/Android). This presents the Menu choices and Media Lists from the Russound audio devices to a third party UI via a small set of RIO Media Management (MM) events. These events include support for menu browsing in several formats including JSON.
- Support for submitting multiple commands in a single request (with multiple results in a single response)
- Easy-to-understand error responses for illegal or malformed commands

*Note: Phones or tablets that use our **MyRussound™** app (iOS/Android) use an IP connection, but the connection to the controller is only maintained while the app is open. Because the XTS is designed to be an always available control point, it keeps the IP connection to its controller open all of the time. Third party integration should take into consideration the limited number of simultaneous RIO connections, and only maintain the connection while the control system is being used.*

Conventions

This section describes conventions used throughout this document. Note that these document conventions are used for clarification only. The RIO Protocol is entirely case-insensitive. To be more specific, the case of incoming commands are ignored. Outgoing responses are formatted exactly as they appear in this document, regardless of the case of the incoming command.

RIO commands are specified in all uppercase. For example,

```
GET C[1].Z[4].currentSource
```

In this case, 'GET' is a RIO command.

Text enclosed in '< >' provides a short description of variable text data that is part of the RIO message. For example,

```
S VERSION="<version #>"
```

In this case, 'version #' is a short description that is replaced with the actual RIO version number at runtime.

"<key>" refers to the RIO command key string.

The key syntax is described in detail in the "RIO Key String" section at the end of this document.

Command Overview

RIO defines a small set of commands, providing access to many of the capabilities of the Russound System. These commands are presented here in general terms, along with a description of the relevant parameters. All commands are available via serial and IP.

VERSION - request the version of the supported RIO protocol

GET - return the value of one or more system parameters

SET - modify the value of one or more system parameters

ADJUST - modify the relative value of one or more system parameters

EVENT - send an event from a zone

WATCH - register to receive asynchronous messages from a particular zone, source or system

RIO Protocol Syntax

This section presents each RIO Command and Response in detail.

The specifics of the RIO protocol are:

RIO commands are case-insensitive.

RIO commands are made up of ASCII characters except for the terminating characters.

All RIO commands must be terminated with a <CR> (0x0D hex)

All RIO responses are terminated with a <CR><LF> (0x0D 0x0A hex)

RIO Command Response Syntax

For RIO commands that are processed successfully, a response is sent with this format:

```
S <optional data>
```

For RIO commands that result in failure, a response is sent with this format:

```
E <error message>
```

For asynchronous RIO responses, or ‘notifications’, a response is sent with this format:

```
N <key>=”<value>”
```

The ‘VERSION’ Command

The *VERSION* command is used to request the version of the RIO protocol running on the device.

VERSION Command Syntax:

```
VERSION
```

VERSION Response:

```
S VERSION=“<version #>”
```

VERSION Examples

1) Request the RIO Protocol version:

VERSION Command:

```
VERSION
```

VERSION Response:

```
S VERSION= “01.16.00”
```

The ‘GET’ Command

The *GET* command returns the value of one or more system parameters. This command is performed synchronously, returning the current value once upon request. A system parameter is addressed by a ‘key’ string. For more details on the ‘key’ string, see the section at the end of this document.

To request a single system parameter,

GET Command Syntax:

```
GET <key>
```

Successful Response:

```
S <key>=”<value>”
```

To request multiple system parameters,

GET Command Syntax:

```
GET <key1>, <key2>, ... , <keyN>
```

Successful Response:

```
S <key1>=”<value1>”, <key2>=”<value2>”, ... , <keyN>=”<valueN>”
```

The tables below provide a complete list of the 'key' strings supported by the *GET* command.

Controller GET Key Table

Controller GET Key Table		
Controller table indices are identified by '*c*', a number from 1 to 6. For MBX and XStream Series, this value is always '1' (i.e. C[1].ipAddress).		
Key	**Description**	**Range**
C[c].ipAddress	IP Address for the controller	a valid IP Address (xxx.xxx.xxx.xxx)
C[c].macAddress	MAC Address for the controller	a valid MAC Address (xx:xx:xx:xx:xx:xx)
C[c].type	Controller type (i.e., model)	MBX-Series: • MBX-PRE • MBX-AMP MCA-Series: • MCA-66 • MCA-88 • MCA-88X XStream Series: • XSource • XZone4 • XZone70V *Note: Version 1.09.00 reports:* *MCA-Series: "MCA-C6"* *XStream Series: "XStream-X5"*
C[c].firmwareVersion	Controller firmware version	Firmware version (xx.xx.xx)

System GET Key Table

System GET Key Table		
Key	**Description**	**Range**
System.status	System status	OFF/ON *Not supported on MBX and XStream Series*
System.language	System Language	ENGLISH/CHINESE/RUSSIAN
System.favorite[f].valid	Indicates whether a favorite is in use. ‘f’ is the favorite number	TRUE/FALSE f = 1..32
System.favorite[f].name	Name of the favorite ‘f’ is the favorite number	50 Char Max f = 1..32
System.Support.grouping	Indicates whether or not device grouping is supported.	“TRUE” – Device grouping is supported. “FALSE” – Device grouping is not supported. Note: Error is an indication of no support for device grouping in RIO.
System.Support.alarm	Indicates whether or not the device supports an alarm.	“TRUE” – Alarm is supported. “FALSE” – Alarm is not supported. Note: Error is an indication of no alarm support in RIO.
System.Support.sleepTime	Indicates whether or not the device supports direct setting of sleep time.	“TRUE” – Sleep time is supported. “FALSE” – Sleep time is not supported. Note: Error is an indication of no sleep time support in RIO.

System GET Key Table		
Key	**Description**	**Range**
System.Support.favoritesV2	Indicates whether or not the device supports version 2 enhanced favorite features	"TRUE" – V2 favorite features are supported. "FALSE" – V2 favorite features are supported. not supported. Note: Error is an indication of no V2 favorite features support in RIO. V2 features inclue - source, sourceType, providerMode, albumCoverURL, playerData
Notes:	System.Grouping is available if System.Support.grouping = "TRUE"	
System.Grouping.deviceList	List of devices on the network capable of grouping.	Use "WATCH Grouping" for deviceList updates. Read-only string.
System.Grouping.uuid	Unique identifier for the connected device.	Read-only string

Zone Get Key Table

Zone GET Key Table

Controller table indices are identified by ‘*c*’, a number from 1 to 6.

Zone table indices are identified by ‘*z*’, a number from 1 to <maxZones>.

<maxZones>: MCA-66 (6), MCA-88 (8), MCA-88X (8)

For MBX and XStream Series, these values are always ‘1’ (i.e. C[1].Z[1].name).

Key	**Description**	**Range**
C[*c*].Z[*z*].name	Name of the specified zone	37 char max
C[*c*].Z[*z*].currentSource	Current physical Source selection for the zone	1 to <maxSources> MCA-66 (6) MCA-88, MCA-88X (8) MBX-Series (1)
C[*c*].Z[*z*].volume	Volume setting for the zone	0 to 50
C[*c*].Z[*z*].bass	Bass setting for the zone	-10 to 10
C[*c*].Z[*z*].treble	Treble setting for the zone	-10 to 10
C[*c*].Z[*z*].balance	Balance setting for the zone	-10 to 10
Notes:	*Note: For the ‘balance’ parameter, a value of ‘-10’ represents the leftmost position in the stereo spectrum, 0 represents the center and ’10’ represents the rightmost position.*	
C[*c*].Z[*z*].loudness	Loudness setting for the zone	OFF/ON
C[*c*].Z[*z*].turnOnVolume	Turn On Volume setting for the zone	0 to 50
C[*c*].Z[*z*].doNotDisturb	Do Not Disturb setting for the zone	OFF/ON/SLAVE
C[*c*].Z[*z*].partyMode	Party Mode setting for the zone	OFF/ON/MASTER

Zone GET Key Table

Controller table indices are identified by ‘*c*’, a number from 1 to 6.

Zone table indices are identified by ‘*z*’, a number from 1 to <maxZones>.

<maxZones>: MCA-66 (6), MCA-88 (8), MCA-88X (8)

For MBX and XStream Series, these values are always ‘1’ (i.e. C[1].Z[1].name).

Key	**Description**	**Range**
C[c].Z[z].status	Power status for the zone	OFF/ON
C[c].Z[z].mute	Mute status for the zone	OFF/ON
C[c].Z[z].sharedSource	Shared source status for the zone	OFF/ON
C[c].Z[z].lastError	Error string triggered by the last zone operation	12 char max
C[c].Z[z].page	Page status for the zone	OFF/ON
C[c].Z[z].sleepTimeDefault	Initial set play time before audio is paused or zone is turned off.	15 minutes
C[c].Z[z].sleepTimeRemaining	Remaining time before audio is paused or zone is turned off.	0 to 60 minutes
C[c].Z[z].favorite[f].valid	Indicates whether a favorite is in use. ‘f’ is the favorite number	TRUE/FALSE f = 1..2
C[c].Z[z].favorite[f].name	Name of the favorite ‘f’ is the favorite number	“F1” or “F2” f = 1..2
C[c].Z[z].enabled	Indicates whether a zone is available	TRUE/FALSE
C[c].Z[z].S[s].enabled	Indicates whether a source is available to a zone	TRUE/FALSE S = 1..<maxSources>

Source GET Key Table

Source GET Key Table		
Source table indices are identified by ‘*s*’, a number from 1 to <maxSources>.		
Key	**Description**	**Range**
S[s].name	Name of the specified source	24 char max
S[s].type	Type of the specified source. *Note: Russound Media Streamer refers to the following:* • *DMS-3.1 Streamer (discontinued)* • *XStream-X5 (discontinued)* • *MCA-88X Internal Streamer* • *XSource, XZone4 and XZone70V* • *MBX-PRE and MBX-AMP*	Amplifier Television Cable Video Accessory Satellite VCR Blu-ray / DVD Receiver Miscellaneous Audio CD Home Control Russound Media Streamer Russound DMS-3.1 AM/FM Tuner Russound ST-1 AM/FM Tuner Russound Bluetooth Module

<table>
<tr><th colspan="3">Source GET Key Table</th></tr>
<tr><th colspan="3">Source table indices are identified by ‘<i>s</i>’, a number from 1 to <maxSources&gt;.</th></tr>
<tr><th>Key</th><th>Description</th><th>Range</th></tr>
<tr><td><i>Notes:</i></td><td colspan="2">A source that is not configured will return “Misc Audio” (miscellaneous audio) as its type value.
For example, if Source 4 has not been configured for use,
sending:
• GET S[4].type
will produce this response:
• S S[4].type="Misc Audio"</td></tr>
<tr><td>S[s].channel</td><td>Channel (frequency) of the Now Playing content on the specified source.</td><td>37 char max</td></tr>
<tr><td><i>Notes:</i></td><td colspan="2">The ‘channel‘ key is valid for these source types:
• DMS-3.1 AM/FM Tuner
• ST-1 AM/FM Tuner</td></tr>
<tr><td>S[s].coverArtURL</td><td>Cover Art URL for the now playing content on the specified source.</td><td>255 char max</td></tr>
<tr><td><i>Notes:</i></td><td colspan="2">The ‘coverArtURL’ key is valid only for Russound Media Streamer</td></tr>
<tr><td>S[s].channelName</td><td>Channel name of the now playing content on the specified source.</td><td>37 char max</td></tr>
<tr><td><i>Notes:</i></td><td colspan="2">The ‘channelName‘ key is valid only for Russound Media Streamer</td></tr>
<tr><td>S[s].genre</td><td>Genre of the now playing content on the specified source.</td><td>37 char max</td></tr>
</table>

Source GET Key Table		
Source table indices are identified by ‘*s*’, a number from 1 to <maxSources>.		
Key	**Description**	**Range**
Notes:	The ‘genre’ key is currently not implemented.	
S[s].artistName	Artist of the now playing content on the specified source.	37 char max
Notes:	The ‘artistName’ key is valid only for Russound Media Streamer	
S[s].albumName	Album for the now playing content on the specified source.	37 char max
Notes:	The ‘albumName‘ key is valid only for Russound Media Streamer	
S[s].playlistName	Playlist for the now playing content on the specified source.	37 char max
Notes:	The ‘playlistName‘ key is valid only for Russound Media Streamer	
S[s].songName	Song for the now playing content on the specified source.	37 char max
Notes:	The ‘songName’ key is valid only for Russound Media Streamer	
S[s].programServiceName	Program Service Name (PSN) of the now playing content on the specified source.	37 char max

Source GET Key Table

Source table indices are identified by ‘*s*’, a number from 1 to <maxSources>.

Key	Description	Range
Notes:	The ‘programServiceName’ key is valid for these source types: • DMS-3.1 AM/FM Tuner • ST-1 AM/FM Tuner	
S[s].radioText	First line of radio text for the now playing content on the specified source.	37 char max
Notes:	The ‘radioText’ key is valid for these source types: • DMS-3.1 AM/FM Tuner • ST-1 AM/FM Tuner	
S[s].shuffleMode	Shuffle mode for the now playing content on the specified source.	OFF/ON
Notes:	The ‘shuffleMode’ key is valid only for Russound Media Streamer	
S[s].repeatMode	Repeat mode for the now playing content on the specified source.	OFF/SINGLE/ALL
Notes:	The ‘repeatMode’ key is valid only for Russound Media Streamer	

<table>
<tr><th colspan="3">Source GET Key Table</th></tr>
<tr><td colspan="3">Source table indices are identified by ‘<i>s</i>’, a number from 1 to <maxSources&gt;.</td></tr>
<tr><th>Key</th><th>Description</th><th>Range</th></tr>
<tr><td>S[s].mode</td><td>Provider mode or streaming service for the now playing content on the specified source.
NOTE: vTuner service is indicated by the value “Internet Radio”.</td><td>Unknown
AirPlay
Spotify
Pandora
SiriusXM
TuneIn
Internet Radio
Media Server
USB
Airable Radio
Deezer
Tidal
Napster
Chromecast
Bluetooth</td></tr>
<tr><td><i>Notes:</i></td><td colspan="2">The ‘mode’ key is valid only for Russound Media Streamer</td></tr>
<tr><td>S[s].Support.MM.longList</td><td>Source supports MM Long List commands</td><td>TRUE/FALSE
Error indicates non-support</td></tr>
<tr><td><i>Notes:</i></td><td colspan="2">The ‘Support.MM.longList’ key is valid only for Russound Media Streamer</td></tr>
</table>

Source GET Key Table		
Source table indices are identified by ‘*s*’, a number from 1 to <maxSources>.		
Key	**Description**	**Range**
S[s].Support.MM.restoreOption	Source supports restoreOption feature	TRUE/FALSE Error indicates non-support See “Select Menu Option” section.
Notes:	The ‘Support.MM.restoreOption’ key is valid only for Russound Media Streamer	
S[s].Support.availableControls	Source supports availableControls feature	TRUE/FALSE Error indicates non-support
Notes:	The ‘Support.availableControls’ key is valid only for Russound Media Streamer	
S[s].Support.trackTime	Source supports trackTime feature	TRUE/FALSE Error indicates non-support
Notes:	The ‘Support.trackTime’ key is valid only for Russound Media Streamer	
S[s].Support.playTime	Source supports playTime feature	TRUE/FALSE Error indicates non-support
Notes:	The ‘Support.playTime key is valid only for Russound Media Streamer	

Source GET Key Table		
Source table indices are identified by ‘*s*’, a number from 1 to <maxSources>.		
Key	**Description**	**Range**
S[s].Support.playerData	Source supports playerData feature	TRUE/FALSE Error indicates non-support
Notes:	The ‘Support.playerData key is valid only for Russound Media Streamer	
S[s].Support.favoritesV2	Source supports favoritesV2	TRUE/FALSE Error indicates non-support
Notes:	The ‘Support.favoritesV2 key is valid only for Russound Media Streamer	
S[s].rating	Rating for the now playing content on the specified source.	UNKNOWN/LIKE/DISLIKE
Notes:	The ‘rating’ key is valid only for Russound Media Streamer	
S[s].playStatus	Current play status	playing paused stopped transitioning
S[s].availableControls	Playback controls that are available for the current play mode and playback state.	JSON string

<table>
<tr><th colspan="3">Source GET Key Table</th></tr>
<tr><th colspan="3">Source table indices are identified by ‘*s*’, a number from 1 to <maxSources>.</th></tr>
<tr><th>Key</th><th>Description</th><th>Range</th></tr>
<tr><td>S[s].sampleRate</td><td>Sample rate of the currently playing item</td><td>0 - 192000</td></tr>
<tr><td>S[s].bitRate</td><td>Sample rate of the currently playing item</td><td>0 - 999999</td></tr>
<tr><td>S[s].bitDepth</td><td>Sample rate of the currently playing item</td><td>0 - 32</td></tr>
<tr><td>S[s].playTime</td><td>Play time into the current track in seconds.</td><td>0 - n</td></tr>
<tr><td>S[s].trackTime</td><td>Total track time in seconds.</td><td>0 - n</td></tr>
<tr><td>S[s].playerData</td><td>Data that can be saved and re-written to restore playback</td><td>JSON string</td></tr>
<tr><td>*Notes:*</td><td colspan="2">When re-applying playerData using RestorePlayerData event, any quotes within the JSON string must be escaped with ‘\’ character. Only media players that support RIO v1.15 and above can use this feature.</td></tr>
</table>

Source GET Key Table (Banks and Presets)

<table>
<tr><th colspan="3">Source GET Key Table (Banks and Presets)</th></tr>
<tr><td colspan="3">Source table indices are identified by ‘<i>s</i>’, a number from 1 to <maxSources>.
Bank ‘b’ is a number from 1 to 6
Preset ‘p’ is a number from 1 to 6
Banks and Preset are valid for AM/FM Tuner types only.</td></tr>
<tr><th>Key</th><th>Description</th><th>Range</th></tr>
<tr><td>S[s].B[b].name</td><td>Name of bank</td><td>Max 12 chars</td></tr>
<tr><td>S[s].B[b].P[p].name</td><td>Name of preset</td><td>Max 12 chars</td></tr>
<tr><td>S[s].B[b].P[p].valid</td><td>Indicate whether preset is in use</td><td>TRUE / FALSE</td></tr>
</table>

GET Examples

1) Get the value for the current source of controller 1, zone 4:

GET Command:

```
GET C[1].Z[4].currentSource
```

GET Response:

```
S C[1].Z[4].currentSource=”1”
```

2) Get the values for the bass and treble of controller 1, zone 4:

GET Command:

```
GET C[1].Z[4].bass, C[1].Z[4].treble
```

GET Response:

```
S C[1].Z[4].bass=”6”, C[1].Z[4].treble=”5”
```

3) Get the IP Address of controller 1:

GET Command:

```
GET C[1].ipAddress
```

GET Response:

```
S C[1].ipAddress=”192.168.1.10”
```

The ‘SET’ Command

The *SET* command changes one or more system parameters. A system parameter is addressed by a ‘key’ string. For more details on the ‘key’ string, see the section at the end of this document.

SET operations are not subject to the current state of the system. They may be utilized at any time, provided the controller/zone is present in the system.

To modify a single system parameter,

SET Command Syntax:

```
SET <key>=”<value>”
```

Successful Response:

```
S <key>=”<value>”
```

To modify multiple system parameters,

SET Command Syntax:

```
SET <key1>=”<value1>”, <key2>=”<value2>”, ..., <keyN>=”<valueN>”
```

Successful Response:

```
S <key1>=”<value1>”, <key2>=”<value2>”, ..., <keyN>=”<valueN>”
```

A successful response returns the modified value.

The tables below provide a complete list of the ‘key’ strings supported by the *SET* command.

System SET Key Table

System SET Key Table		
Key	**Description**	**Range**
System.language	System Language	ENGLISH/CHINESE/RUSSIAN
System.favorite[f].name	System Favorite Name	Max 50 chars Favorites numbered 1..32

Zone SET Key Table

Zone SET Key Table		
Controller table indices are identified by ‘c’, a number from 1 to 6. Zone table indices are identified by ‘z’, a number from 1 to <maxZones>. <maxZones>: MCA-66 (6), MCA-88 (8), MCA-88X (8) For MBX and XStream Series, these values are always ‘1’ (i.e. C[1].Z[1].bass).		
Key	**Description**	**Range**
C[c].Z[z].bass	Bass setting for the zone	-10 to 10
C[c].Z[z].treble	Treble setting for the zone	-10 to 10
C[c].Z[z].balance	Balance setting for the zone	-10 to 10
Notes:	Note that for the ‘balance’ parameter, a value of ‘-10’ represents the leftmost position in the stereo spectrum, 0 represents the center and ’10’ represents the rightmost position.	
C[c].Z[z].loudness	Loudness setting for the zone	OFF/ON
C[c].Z[z].turnOnVolume	Turn On Volume setting for the zone	0 to 50

The Source table below provides a complete list of the ‘key’ strings supported by the *SET* command for sources. Source tables indices are identified by ‘*s*’, a number from 1 to <maxSources>

Source SET Key Table

<table>
<tr><th colspan="3">Source SET Key Table
Source tables indices are identified by ‘s, a number from 1 to 8.
Bank ‘b’ is a number from 1 to 6
Preset ‘p’ is a number from 1 to 6
Banks and Preset are valid for AM/FM Tuner types only.</th></tr>
<tr><th>Key</th><th>Description</th><th>Range</th></tr>
<tr><td>S[s].B[b].name</td><td>Bank name</td><td>Max 12 chars
Banks are numbered 1..6</td></tr>
<tr><td><i>Notes:</i></td><td colspan="2">See for setting tuner presets</td></tr>
</table>

SET Examples

1) Set the value for the Turn On Volume of controller 1, zone 4:

SET Command:

```
SET C[1].Z[4].turnOnVolume=”25”
```

SET Response:

```
S C[1].Z[4].turnOnVolume=”25”
```

2) Set the values for the bass and treble of controller 1, zone 4:

SET Command:

```
SET C[1].Z[4].bass=”10”, C[1].Z[4].treble=”8”
```

SET Response:

```
S C[1].Z[4].bass=”10”, C[1].Z[4].treble=”8”
```

The ‘ADJUST’ Command

The *ADJUST* command increments or decrements the current value of one or more system parameters by one. A system parameter is addressed by a ‘key’ string. For more details on the ‘key’ string, see the section at the end of this document.

The ADJUST command is useful in implementing controls that are intended to make adjustments to a system parameter relative to their current value.

To modify a single system parameter,

ADJUST Command Syntax:

```
ADJUST <key>=”+1”
ADJUST <key>=”-1”
```

Successful Response:

```
S <key>=”<value>”
```

A successful response returns the modified value.

To modify multiple system parameters,

ADJUST Command Syntax:

```
ADJUST <key1>=”<value1>”, <key2>=”+/-<value2>”, ..., <keyN>=”+/-<value3>”
```

where <valueN> is +1 or -1

Successful Response:

S <key1>="<value1>", <key2>="<value2>", ..., <keyN>="<valueN>"

where <valueN> is the adjusted value of the parameter

A successful response returns the modified value.

The table below provides a complete list of the 'key' strings supported by the *ADJUST* command. It also specifies the allowable adjustment range for each.

Adjustments that result in out-of-range data values will not result in an error. However, the value of the parameter, after executing the ADJUST command, are governed not to exceed the allowable parameter range. These allowable parameter ranges are specified as '**Data Range**' in the table below.

Controller table indices are identified by '*c*', a number from 1 to 6.

Zone table indices are identified by '*z*', a number from 1 to <maxZones>.

<maxZones>: MCA-66 (6), MCA-88 (8), MCA-88X (8)

For MBX and XStream Series, these values are always '1' (i.e. C[1].Z[1].bass).

Key	Description	Range
C[*c*].Z[*z*].bass	Bass setting for the zone	-10 to +10
C[*c*].Z[*z*].treble	Treble setting for the zone	-10 to +10
C[*c*].Z[*z*].balance	Balance setting for the zone	-10 to +10
C[*c*].Z[*z*].turnOnVolume	Turn On Volume setting for the zone	0 to 50

Note that for the 'balance' parameter, a value of '-10' represents the leftmost position in the stereo spectrum, 0 represents the center and '10' represents the rightmost position.

ADJUST Examples

1) Increase the value for the ***turn on volume*** of controller 1, zone 4, currently set to a value of ’20’:

ADJUST Command:

```
ADJUST C[1].Z[4].turnOnVolume=”+1”
```

ADJUST Response:

```
S C[1].Z[4].turnOnVolume=”21”
```

2) For controller 1, zone 1, simultaneously increase the value of the ***bass,*** currently set to a value of ‘1’ and lower the value of the ***treble,*** currently set to a value of ‘-2’:

ADJUST Command:

```
ADJUST C[1].Z[4].bass=”+1”, C[1].Z[4].treble=”-1”
```

ADJUST Response:

```
S C[1].Z[4].bass=”2”, C[1].Z[4].treble=”-3”
```

The ‘EVENT’ Command

The *EVENT* command is typically used to issue commands that are triggered by user actions (i.e., button presses, screen selections, etc). These commands may change system parameter values (such as zone volume adjustments). Unlike the SET and ADJUST commands, the EVENT commands may also affect system state (such as zone on/off status, party mode state, ...), depending on current conditions.

Due to their stateful behavior, executing the same EVENT command can provide different resultant values. For example, setting a controller/zone Party Mode to ‘ON’, when no Party is ongoing, will result in setting that controller/zone to Party Mode ‘MASTER’ status, since Party Mode requires at least one Master controller/zone.

Events are directed at a controller/zone pair and specified by an *Event ID and one or two event-specific data values.*

EVENT Command Syntax:

```
EVENT C[c].Z[z]!<event id> <data1> <data2>
```

Successful Response:

```
S
```

Physical vs Logical Source Selection

Source selection can be performed in two ways; physically and logically.

A physical source selection treats the supplied source number in terms of the source inputs as they appear on the rear panel of the System Controller.

Physical source selection is accessed using this syntax:

```
EVENT C[c].Z[z]!SelectSource <physical source number>
```

A logical source selection ignores the ‘excluded’ and ‘unconfigured’ sources. That is, the available sources (on a per-zone basis) are numbered from 1 to N, where *N* is the total number of available sources. Note that the Russound System Remote Control refers to the sources as ‘logical’ sources.

Logical source selection is accessed using this syntax:

EVENT C[c].Z[z]!KeyRelease SelectSource <logical source number>

Sleep Timer

The sleep timer can be set by one of two ways: If System.Support.sleepTime is set to true, then a value can be sent to adjust the sleep timer timeout:

EVENT C[c].Z[z]!KeyRelease Sleep <timeout>

timeout: sleep timer in minutes, or 0 to disable.

Otherwise, for MCA-Series the Sleep key code can be sent in an event to adjust the sleep timer timeout by iterating through a list of predefined settings (15, 30, 45, 60, 0).

For MBX-Series sending the Sleep key code simply resets the sleep timer timeout to 15 minutes.

EVENT C[c].Z[z]!KeyCode 57

The sleep timer value is returned in C[c].Z[z].sleepTimeRemaining when the event is sent.

EVENT Table

The Events table below provides a complete list of the id's supported by the EVENT command.

Event Table			
Description	**Event ID**	**Data 1** **Range**	**Data 2** **Range**
Reboot the device	Reboot	N/A	N/A
Turn a zone on	ZoneOn	N/A	N/A
Turn a zone off	ZoneOff	N/A	N/A
Turn all zones on	AllOn	N/A	N/A
Turn all zones off	AllOff	N/A	N/A
Mute a zone	ZoneMuteOn	N/A	N/A
Select a physical source	SelectSource	1 to <maxSources>	N/A
Unmute a zone	ZoneMuteOff	N/A	N/A
Send a Key Press	KeyPress	<key code>	N/A (except Volume)
Send a Key Release	KeyRelease	<key code>	N/A (except SelectSource and Sleep)
Send a Key Hold	KeyHold	<key code>	hold time (in msec)
Send a UEI Key Code	KeyCode	<1-100> (see Table of Numeric Key Codes)	N/A
Change Party Mode	PartyMode	off/on/master	N/A
Change DND	DoNotDisturb	off/on	N/A
Change Shuffle Mode	Shuffle	N/A	N/A

Event Table			
Description	**Event ID**	**Data 1 Range**	**Data 2 Range**
Change Repeat Mode	Repeat	N/A	N/A
Set Seek Time	SetSeekTime	seconds into track	N/A
Favorite Related Events			
Save a zone favorite	SaveZoneFavorite	“<fav-name”> (must be quotes)	fav number = 1..2
Save a system favorite	SaveSystemFavorite	“<fav-name”> (must be quotes)	fav number = 1..32
Restore a zone favorite	RestoreZoneFavorite	fav number = 1..2	
Restore a system favorite	RestoreSystemFavorite	fav number = 1..32	
Delete a zone favorite	DeleteZoneFavorite	fav number = 1..2	
Delete a system favorite	DeleteSystemFavorite	fav number = 1..32	
Restore playback using player data from S[s].playerData notification	RestorePlayerData	<quoted json player data> (quotes within json string must be escaped with ‘\’ character)	
Preset Related Events (For use with Russound ST-1, ST-2, MCA-C5 internal tuner and DMS-3.1 internal tuner)			
Save a preset w/ optional name	SavePreset	“<preset-name”> (must be quoted)	preset number = 1..36
Restore a preset	RestorePreset	preset number = 1..36	
Delete a preset	DeletePreset	preset number = 1..36	

Event Table			
Description	**Event ID**	**Data 1** **Range**	**Data 2** **Range**
Grouping Related Events			
Group with a device with the following uuid	Group	“<uuid>” Device uuid can be determined by parsing through System. Grouping.deviceList	N/A
Detach from group	Ungroup	N/A	N/A

Key Events

Key Events are intended to emulate all of the buttons available on the Russound System Remote Control. They provide the Press/Release/Hold conditions, where applicable. The following tables list the EVENT Key Codes for each of these conditions.

Note that for Media Management source types, certain Key Codes – generally those that manage ‘transport’ functions such as ‘Play’, ‘Pause’, ‘Next’, etc. – are only processed after the source has been properly initialized. Please refer to the **Media Management** section for details.

KeyPress Events

The table below lists the key codes that are supported by the **KeyPress** EVENT

RIO Key Codes
Volume (0 to 50)
VolumeUp
VolumeDown
GroupVolume (0 to 50)

KeyRelease Events

The table below lists the key codes that are supported by the **KeyRelease** EVENT. Entries identified with a ‘*’ are configurable via the Web Config Command Editor (applicable to non-Russound sources only).

RIO Key Codes		
DigitZero	*ChannelUp	MenuLeft
DigitOne	*ChannelDown	MenuRight
DigitTwo	Power	MenuUp
DigitThree	*Stop	MenuDown
DigitFour	*Pause	Select
DigitFive	Favorite1	Info
DigitSix	Favorite2	Menu
DigitSeven	*Play	Record
DigitEight	Enter	PageUp
DigitNine	Last	PageDown
*Previous	Sleep (timeout 0-60)	Disc
*Next	Guide	Mute
NextSource	Exit	SelectSource (1 to 8, logical)

Note: logical source selection refers to the number as listed (exludes disabled sources).

KeyHold Events

The table below lists the key codes that are supported by the **KeyHold** EVENT.

KeyHold events must be accompanied by a ‘hold time’ parameter. The hold time is specified in milliseconds.

In order for the KeyHold EVENTs to operate correctly, they must be executed in a specific manner by the 3rd party device (the RIO ‘client’). The KeyHold EVENT message must be transmitted once every 150 milliseconds for as long as the button is held. The ‘hold time’ parameter should be increased by 150 each time it is retransmitted. When the button is released, a KeyRelease EVENT command must be sent to complete the hold operation. See EVENT Example #2 for more details.

Entries identified with a ‘*’ are configurable via the Web Config Command Editor (applicable to non-Russound sources only).

RIO Key Codes		
DigitZero	*ChannelDown	MenuLeft
DigitOne	Power	MenuRight
DigitTwo	*Stop	MenuUp
DigitThree	*Pause	MenuDown
DigitFour	Favorite1	Select
DigitFive	Favorite2	Info
DigitSix	*Play	Menu
DigitSeven	Mute	Record
DigitEight	Enter	PageUp
DigitNine	Last	PageDown
*Previous	Sleep	Disc
*Next	Guide	
*ChannelUp	Exit	

EVENT Examples

1) Increase the volume of controller 1, zone 4:

EVENT Command:

```
    EVENT C[1].Z[4]!KeyPress VolumeUp
```

EVENT Response:

```
    S
```

2) Perform a Search Forward for approximately 1 second on an iPod for controller 1, zone 4:

EVENT Commands:

```
        <user presses key>
        <150msec time delay>
    EVENT C[1].Z[4]!KeyHold Next 150
        <150msec time delay>
    EVENT C[1].Z[4]!KeyHold Next 300
        <150msec time delay>
    EVENT C[1].Z[4]!KeyHold Next 450
        <150msec time delay>
    EVENT C[1].Z[4]!KeyHold Next 600
        <150msec time delay>
    EVENT C[1].Z[4]!KeyHold Next 750
        <150msec time delay>
    EVENT C[1].Z[4]!KeyHold Next 900
        <150msec time delay>
    EVENT C[1].Z[4]!KeyHold Next 1050
        <user releases key>
    EVENT C[1].Z[4]!KeyRelease Next
```

EVENT Response:

```
    S <for each EVENT received>
```

AM/FM Tuner Specific Key Events

tunerBandToggle	"KeyRelease Play"
tunerMuteToggle	"KeyRelease Pause"
tunerUp	"KeyRelease ChannelUp"
tunerDown	"KeyRelease ChannelDown"
tunerPresetNext	"KeyRelease Next"
tunerPresetPrev	"KeyRelease Previous"
tunerBankNext	"KeyRelease PageUp"
tunerBankPrev	"KeyRelease PageDown"
tunerSeek	"KeyCode 27"
tunerScan	"KeyCode 70"
tunerStop	"KeyCode 58"
tunerStereo	"KeyCode 51"
tunerMono	"KeyCode 52"
tunerLocal	"KeyCode 66"
tunerDistant	"KeyCode 67"

Managing Favorites

RIO supports management of system and zone favorites. This is done through a combination of events as shown in the table listed in the EVENTS section and GET operations as listed in the tables in the ‘GET’ section.

MCA-Series, XStream Series and MBX Series allow 32 system-wide favorites and 2 additional favorites per zone. Each favorite has a name and a flag indicating whether or not it is currently in use (valid). Use WATCH SYSTEM to receive notification of changes to system favorites. Also, these properties may be examined (but not set) through the GET command as follows:

Note: The allowable string size for naming favorites is 50 characters

System Favorites:

GET System Favorite Valid

```
GET system.favorite[8].valid
S System.favorite[8].valid="FALSE"
```

GET System Favorite Name

```
GET system.favorite[8].name
S System.favorite[8].name="Favorite #8"
```

SET System Favorite Name

```
SET system.favorite[8].name=”My Favorite”
S System.favorite[8].name=" My Favorite "
```

GET System Favorite Source

```
GET system.favorite[8].source
S System.favorite[8].source="1"
```

GET System Favorite sourceType

```
GET system.favorite[8].sourceType
```

```
S System.favorite[8].sourceType="Russound Media Streamer"
```

GET System Favorite providerMode

```
GET system.favorite[8].providerMode
S System.favorite[8].providerMode="Tidal"
```

GET System Favorite albumCoverURL

```
GET system.favorite[8].albumCoverURL
S System.favorite[8].albumCoverURL="<album cover url>"
```

GET System Favorite PlayerData

```
GET system.favorite[8].playerData
S System.favorite[8].playerData="<JSON player data string>"
```

Zone Favorites:

GET Zone Favorite Valid

```
GET C[1].Z[1].favorite[2].valid
S C[1].Z[1].favorite[2].valid="FALSE"
```

GET Zone Favorite Name

```
GET C[1].Z[1].favorite[2].name
S C[1].Z[1].favorite[2].name="F2"
```

GET Zone Favorite Source

```
GET C[1].Z[1].favorite[2].source
S C[1].Z[1].favorite[2].source="1"
```

GET Zone Favorite sourceType

```
        GET C[1].Z[1].favorite[2].sourceType
        S C[1].Z[1].favorite[2].sourceType="Russound Media Streamer"
```

GET Zone Favorite providerMode

```
        GET C[1].Z[1].favorite[2].providerMode
        S C[1].Z[1].favorite[2].providerMode="Tidal"
```

GET Zone Favorite albumCoverURL

```
        GET C[1].Z[1].favorite[2].albumCoverURL
        S C[1].Z[1].favorite[2].albumCoverURL="<album cover url>"
```

GET Zone Favorite PlayerData

```
        GET C[1].Z[1].favorite[2].playerData
        S C[1].Z[1].favorite[2].playerData="<JSON player data string>"
```

Since setting the name of a favorite and making it active are two separate operations that must be performed in the proper order, setting a favorite is accomplished through an event.

```
    EVENT C[1].Z[1]!saveZoneFavorite   "ZoneFav" 1
    S
    EVENT C[1].Z[1]!saveSystemFavorite "SysFav#1" 1
    S
```

Restoring and deleting favorites is straightforward using the events below. They can also be found in the Event table.

```
EVENT C[c].Z[z]!KeyRelease restoreZoneFavorite <fav number 1-2>
S
EVENT C[c].Z[z]!KeyRelease restoreSystemFavorite <fav number 1-32>
```

S

EVENT C[c].Z[z]!KeyRelease deleteZoneFavorite <fav number 1-2>

S

EVENT C[c].Z[z]!KeyRelease deleteSystemFavorite <fav number 1-32>

S

Managing Banks and Presets

RIO supports management of banks and presets. This is done through a combination of events as shown in the table listed in the EVENTS section and GET operations as listed in the tables in the 'GET' section.

Banks and presets operate on a per-source basis, and are only present for source types that support them, which are typically tuners.

Presets are organized into 6 banks of 6 presets each.

Bank name, Preset names and validities are accesses directly through the source with the GET command:

```
SET S[1].B[1].name="MyBank1"
S S[1].B[1].name="MyBank1"

GET S[1].B[1].name
S S[1].B[1].name="MyBank1"

GET S[1].B[1].P[2].valid
S S[1].B[1].P[2].valid="FALSE"

GET S[1].B[1].P[2].name
S S[1].B[1].P[2].name="Preset 2"
```

Note that only the Bank Name is settable through the SET command – banks are always enabled. Individual presets are managed with the EVENT command and are sent to a particular zone. They will affect the source that the zone is currently using.

When managing preset with events, it is necessary to convert the Bank and Preset numbers to a single value between 1 and 36 as follows:

Bank/Preset	Numerical value
Bank 1, Preset 1	1
Bank 1, Preset 6	6
…	
Bank 2, Preset 1	7
Bank 2, Preset 6	12
…	
Bank 6, Preset 1	31
Bank 6, Preset 6	36

To save (enable) a preset with the default name, which is the station frequency, do:

```
EVENT C[1].Z[1]!savePreset    1

S

GET S[1].B[1].P[1].name

S S[1].B[1].P[1].name="89.1 MHz FM"
```

(this assumes that controller 1, zone 1 is tuned to source 1.)

To save with a user-defined name:

```
EVENT C[1].Z[1]!savePreset   "P1" 1

S

GET s[1].B[1].P[1].name

S S[1].B[1].P[1].name="P1"
```

Restoring and deleting presets is straightforward using the events from the Event table.

The ‘WATCH’ Command

The *WATCH* command enables a device to register for and receive asynchronous notifications of system parameter changes. The *WATCH* command groups the system parameters into categories: ZONE, SOURCE and SYSTEM.

Enabling a WATCH on a zone is particularly powerful. In addition to receiving notifications on zone parameters (such as status, volume, etc), the RIO client will also receive notifications for changes to the current source and its parameters (such as songName, shuffleMode, etc).

The WATCH command is an excellent way for a UI device to remain aware of system status, allow the device to display current information with minimal communication or overhead. Each notification message is uniquely identified with a key string. This allows the UI device a way to identify relevant data, while filtering unneeded data. These UI device decisions are sometimes made on a screen-by-screen basis, or a stateful manner.

When the command is issued with the parameter set to ‘ON’, it will provide a snapshot of the system parameters in the requested category. Subsequent changes will be sent to the requesting device as their values change. These asynchronous change notifications will continue until the WATCH command is turned ‘OFF’.

WATCH Command Syntax:

Start a Zone WATCH session

```
WATCH C[c].Z[z] ON
```

Start a Source WATCH session

```
WATCH S[s] ON
```

Start a System WATCH session

```
WATCH System ON
```

Start a Grouping WATCH session

```
WATCH Grouping ON
```

Successful Response:

```
S
N <key1>=”<value1>”
N <key2>=”<value2>”
...
N <keyN>=”<valueN>”
```

WATCH Command Syntax (continued):

Stop a Zone WATCH session

```
WATCH C[c].Z[z] OFF
```

Stop a Source WATCH session

```
WATCH S[s] OFF
```

Stop a System WATCH session

```
WATCH System OFF
```

Successful Response:

```
S
```

Stop a Grouping WATCH session

```
WATCH Grouping OFF
```

Successful Response:

```
S
```

WATCH Notification Messages

The type of notification messages that result from enabling WATCH vary based on the WATCH parameter and current source type. The following sections list the notifications in these various scenarios.

WATCH System

Notifications:

N System.status=”<value>”

N System.favorite[1..32].valid=”<value>”

N System.favorite[n].name=”<name>”

N System.favorite[1..32].source=”<source number>”

N System.favorite[1..32].sourceType=”<source type>”

N System.favorite[1..32].providerMode=”<provider mode i.e “Tidal” >”

N System.favorite[1..32].albumCoverURL=”<album cover url>”

N System.favorite[1..32].playerData=”<JSON player data string>”

Notification when Adding Favorite

N System.favorite[n].valid=”TRUE”

N System.favorite[n].name=”<name>”

N System.favorite[1..32].source=”<source number>”

N System.favorite[1..32].sourceType=”<source type>”

N System.favorite[1..32].providerMode=”<provider mode i.e “Tidal” >”

N System.favorite[1..32].albumCoverURL=”<album cover url>”

N System.favorite[1..32].playerData=”<JSON player data string>”

Notification when Removing Favorite

N System.favorite[n].valid=”FALSE”

WATCH a Zone

Notifications:

```
N C[c].Z[z].name=”<value>”
N C[c].Z[z].status=”<value>”
N C[c].Z[z].currentSource=”<value>”
N C[c].Z[z].volume=”<value>”
N C[c].Z[z].bass=”<value>”
N C[c].Z[z].treble=”<value>”
N C[c].Z[z].balance=”<value>”
N C[c].Z[z].loudness=”<value>”
N C[c].Z[z].doNotDisturb=”<value>”
N C[c].Z[z].partyMode=”<value>”
N C[c].Z[z].turnOnVolume=”<value>”
N C[c].Z[z].mute=”<value>”
N C[c].Z[z].sharedSource=”<value>”
N C[c].Z[z].lastError=”<value>”
N C[c].Z[z].page=”<value>”
N C[c].Z[z].sleepTimeDefault="15"
N C[c].Z[z].sleepTimeRemaining="0"
```

WATCH a Source

For all source types

Notifications:

N S[s].type=”<value>”

where value is one of these possible source types

- Amplifier
- Television
- Cable
- Video Accessory
- Satellite
- VCR
- Blu-ray / DVD
- Receiver
- Miscellaneous Audio
- CD
- Home Control
- Russound Media Streamer
- Russound DMS-3.1 AM/FM Tuner
- Russound ST-1 AM/FM Tuner
- Russound Bluetooth Module

N S[s].name=”<value>”

WATCH a Source (continued)

For All Russound Media Streamers (DMS-3.1, MCA-88X Internal Streamer, XStream-Series and MBX-Series)

N S[s].mode=”<value>”

Possible values are "Media Server", "USB", "AirPlay", "Spotify", "Chromecast", "Bluetooth", "Pandora", "Tidal", "Napster", "Deezer", "SiriusXM", "Internet Radio", "TuneIn", "Airable Radio", "SPDIF", "Line In", "Party", "Unknown"

N S[s].playlistName=”<value>”

N S[s].channelName=”<value>”

N S[s].artistName=”<value>”

N S[s].albumName=”<value>”

N S[s].songName=”<value>”

N S[s].coverArtURL=”<value>”

N S[s].channelArtURL=”<value>”

WATCH a Source (continued)

Media Streamer Mode Metadata Support Table

The table below defines which metadata fields are available for each mode.

Media Streamer Mode:	Metadata Field						
	playListName	**channelName**	**artistName**	**albumName**	**songName**	**coverArtURL**	**channelArtURL**
Media Server (UpnP / DLNA)	playlist / category/ folder (1)	n/a	artist (2)	album (3)	song (4)	album art	n/a
USB	playlist/category/folder (1)	n/a	artist (2)	album (3)	song (4)	album art	n/a
AirPlay	“Airplay” (1)	n/a	artist (2)	album (3)	song (4)	album art	n/a
Spotify	playlist / category/ folder (1)	n/a	artist (2)	album (3)	song (4)	album art	n/a
Chromecast	content description (1)	n/a	artist (2)	album (3)	song (4)	album art	n/a
Bluetooth	“Bluetooth” (1)	n/a	artist (2)	album (3)	song (4)	album art	n/a
Pandora	“Pandora” (1)	station	artist (2)	album (3)	song (4)	album art	n/a
TIDAL	playlist / category / folder (1)	n/a	artist (2)	album (3)	song (4)	album art	n/a
Napster	playlist / category/folder (1)	n/a	artist (2)	album (3)	song (4)	album art	n/a
Deezer	playlist / category / folder (1)	n/a	artist (2)	album (3)	song (4)	album art	n/a
SiriusXM	category / folder (1)	station (2)	artist	album (3)	song (4)	album art	station art *
Internet Radio (vTuner)	category / folder (1)	station (2)	artist (3)	n/a	song (4)	album art	station art *
TuneIn	category / folder (1)	station (2)	artist (3)	n/a	song (4)	album art	station art *
Airable Radio	category / folder (1)	station (2)	artist (3)	n/a	song (4)	album art	station art *
Line In	“Analog Input” (1)	n/a	n/a	n/a	n/a	n/a	n/a
SPDIF	“Digital Input” (1)	n/a	n/a	n/a	n/a	n/a	n/a
Party (Xstream-Series)	“Party” (1)	n/a	artist (2)	album (3)	song (4)	n/a	n/a
Unknown	category / folder (1)	n/a	artist (2)	album (3)	song (4)	album art	n/a

(1)– (4): Recommended order (usually top to bottom) in which text fields appear on user interface.

* Station art should be shown if no album art is available.

WATCH a Source (continued)

N S[s].shuffleMode=”<value>”

Possible values are ‘OFF’ and ‘ON’.

N S[2].repeatMode="<value>"

Possible values are ‘OFF’, ‘SINGLE’ and ‘ALL’.

N S[s].rating="UNKNOWN"

Rating may also have the value LIKE or DISLIKE

N S[s].support.MM.longList="<value>"

Indicates if feature is supported. Possible values are ‘TRUE’, ’FALSE’. (no notification should be interpreted as FALSE.)

N S[s].Support.MM.restoreOption="<value>"

Indicates if feature is supported. Possible values are ‘TRUE’, ’FALSE’. (no notification should be interpreted as FALSE.)

WATCH a Source (continued)

N S[s].Support.availableControls="<value>"

Indicates if feature is supported. Possible values are ‘TRUE’, ’FALSE’. (no notification should be interpreted as FALSE.)

N S[s].Support.trackTime="<value>"

Indicates if feature is supported. Possible values are ‘TRUE’, ’FALSE’. (no notification should be interpreted as FALSE.)

N S[s].Support.playTime="<value>"

Indicates if feature is supported. Possible values are ‘TRUE’, ’FALSE’. (no notification should be interpreted as FALSE.)

N S[s].Support.playerData="<value>"

Indicates if feature is supported. Possible values are ‘TRUE’, ’FALSE’. (no notification should be interpreted as FALSE.)

N S[s].Support.favoritesV2="<value>"

Indicates if feature is supported. Possible values are ‘TRUE’, ’FALSE’. (no notification should be interpreted as FALSE.)

WATCH a Source (continued)

For Russound MBX Streamers

N S[s].trackTime="<value>"

N S[s].playTime="<value>"

N S[s].format="<value>"

N S[s].bitRate="<value>"

N S[s].bitDepth="<value>"

N S[s].playStatus="<value>"

For playStatus, valid values are “playing”, “paused”, “stopped” and “transitioning”

N S[s].availableControls=”<value>”

For availableControls, values are in JSON format

Example:

N S[1].availableControls=
"{"contextMenu":false,"dislike":false,"like":false,"next":false,"pause":false,"play":true,"previous":false,"repeat":false,"seekBack":false,"seekForward":false,"seekTime":false,"seekTrack":false,"shuffle":false,"stop":true}"

N S[1].playerData=”<JSON player data string>”

For Russound AM/FM Tuner Sources

N S[s].channel=”<value>”

N S[s].programServiceName=”<value>”

N S[s].radioText=”<value>”

N S[s].channel=”<value>”

WATCH Examples

1) WATCH for asynchronous changes on Controller 1, Zone 4:

WATCH Command:

```
WATCH C[1].Z[4] ON
```

WATCH Response:

```
S

N C[1].Z[4].status=”ON”
N C[1].Z[4].volume=”20”
N C[1].Z[4].bass=”10”
N C[1].Z[4].treble=”10”
N C[1].Z[4].balance=”10”
N C[1].Z[4].loudness=”OFF”
N C[1].Z[4].currentSource=”2”
N S[2].artistName=”The Beatles”
N S[2].albumName=”Abbey Road”
N S[2].songName=”Come Together”
```

WATCH related notifications:

A song begins to play on the Zone 4 current source...

```
N S[2].artistName=”ABBA”
```

```
N S[2].albumName=”Arrival”
N S[2].songName=”Dancing Queen”
```

Volume is adjusted on Zone 4...

```
N C[1].Z[4].volume=”21”
```

WATCH GROUPING

WATCH Command:

WATCH GROUPING ON

WATCH Response:

S

Notifications for MBX-Series:

N System.Grouping.deviceList=”<value>”

N System.Grouping.uuid=”<value>”

Notifications for MCA-Series:

N System.Grouping.mcaZoneList=”<value>”

N System.Grouping.uuid=””

System.Grouping.deviceList is a JSON formatted string. The table below lists the key-value pairs:

System.Grouping.deviceList			
Key	**Description**	**Value**	
groupId	Group identifier	Not Grouped	*null*
		Group Leader	<uuid>
		Group Follower	<uuid$_{\text{LEADER}}$>
ipAddress	Device IP address	String (xxx.xxx.xxx.xxx)	
name	Device name	String	
uuid	Device unique identifier	String	
volume	Device volume	String “0” - “50”	
playStatus	Play status	“stopped” | “paused” | “playing”	
isGroupable	Grouping is allowed	true | false (Boolean)	
groupVolume	Group volume: Volume changes affect the whole group	Not Grouped	*null*
		Group Leader	String “0” - “50”
		Group Follower	*null*

Below is an example deviceList string with two devices on the network:

```
{"devices":[{
"groupId" : null,
"ipAddress" : "192.168.10.119",
"name" : "Room 1",
"uuid" : "russound-musicbox-0001",
"volume" : "28",
"playStatus" : "stopped",
"isGroupable" : true,
"groupVolume" : null
},{
"groupId" : null,
"ipAddress" : "192.168.10.120",
"name" : "Room 2",
"uuid" : "russound-musicbox-0002",
"volume" : "22",
"playStatus" : "playing",
"isGroupable" : true,
"groupVolume" : null }]}
```

System.Grouping.mcaZoneList is a JSON formatted string. The table below lists the key-value pairs:

System.Grouping.mcaZoneList			
Key	**Description**	**Value**	
groupId	Group identifier	Not Grouped	*null*
		Group Leader	<uuid>
		Group Follower	<uuid$_{\text{LEADER}}$>
ipAddress	Device IP address	String (xxx.xxx.xxx.xxx)	
name	Device name	String	
uuid	Device unique identifier	String	
volume	Device volume	0-50	
status	Zone status	“ON” | “STANDBY”	
sourceId	Source ID number	1-<maxSources>	
groupVolume	Group volume: Volume changes affect the whole group	Not Grouped	*null*
		Group Leader	0-50
		Group Follower	*null*

Below is an example mcaZoneList for the MCA-66 with zones 1 and 2 grouped:

```
{"mcaZones":[
{ "groupId" : "MCA66_C[1].Z[1]",
"ipAddress" : "192.168.10.129",
"name" : "Zone 1",
"uuid" : "MCA66_C[1].Z[1]",
"volume" : "28",
"status" : "ON",
"sourceId" : 5,
"groupVolume" : 28
},{
"groupId" : "MCA66_C[1].Z[1]",
"ipAddress" : "192.168.10.129",
"name" : "Zone 2",
"uuid" : " MCA66_C[1].Z[2]",
"volume" : 16,
"status" : "ON",
"sourceId" : 5,
"groupVolume" : null
},{
"groupId" : null,
"ipAddress" : "192.168.10.129",
"name" : "Zone 3",
"uuid" : "MCA66_C[1].Z[3]",
"volume" : "20",
"status" : "STANDBY",
"sourceId" : 1,
"groupVolume" : null
},{
"groupId" : null,
"ipAddress" : "192.168.10.129",
"name" : "Zone 4",
"uuid" : "MCA66_C[1].Z[4]",
"volume" : "18",
"status" : "STANDBY",
"sourceId" : 2,
"groupVolume" : null
},{
"groupId" : null,
"ipAddress" : "192.168.10.129",
"name" : "Zone 5",
"uuid" : "MCA66_C[1].Z[5]",
"volume" : "16",
"status" : "STANDBY",
"sourceId" : 3,
"groupVolume" : null
},{
"groupId" : null,
"ipAddress" : "192.168.10.129",
"name" : "Zone 6",
"uuid" : " MCA66_C[1].Z[6]",
"volume" : 22,
"status" : "ON",
"sourceId" : 4,
"groupVolume" : null }]}
```

Zone Alarms

Alarm Status

The following is the format of the JSON string for alarm status. This string can be observed by watching a zone using the RIO WATCH command, WATCH C[c].Z[z] ON. This string can also be read using the RIO GET command, GET C[c].Z[z].alarm.status.

```
{ "zoneAlarms" :  {
"version" : "1.0",
"alarms" :  [ {
     "valid" : <true | false>,
     "name" : "Alarm 1",
     "id" : <1-15>,
     "status" : "<DISABLED | OFF | ON | SNOOZE>",
     "enable" : <true | false>,
     "schedule" : <true | false>,
     "minute" : <0-1440>,
     "type" : "<NONE | CHIME | FAVORITE>",
     "sourceId" : <1-8>,
     "chimeId" : <1-12>,
     "favoriteId" : <1-32>,
     "volume" : <0 – 50>,
     "rampVolume" : <true | false>,
     "duration" : <0-86400>,
     "snoozeTime" : <0 – 30>,
     "shuffle" : <true | false>,
     "repeat" : <true | false>,
     "days": {
          "once" : <true | false>,
          "monday" : <true | false>,
          "tuesday " : <true | false>,
          "wednesday" : <true | false>,
          "thursday" : <true | false>,
          "friday" : <true | false>,
          "saturday" : <true | false>,
          "sunday" : <true | false>
     }
}]}}
```

Alarm Configuration

The following is the format of the JSON string for alarm options. Alarms are configured using the RIO EVENT command, EVENT C[c].Z[z]!saveAlarm "<configString>". The most recently set configuration can also be read using the RIO GET command, GET C[c].Z[z].alarm.config.

The JSON sting must be escaped when using the RIO EVENT command to configure an alarm. Also, alarm status is not set by alarm configuration. Defaults are indicated by [value].

```
{
"valid" : <true | false> [false],
"name" : "<alarm name>", [""]
"id" : <1-15>, <0 reserved for chime preview>
"enable" : <true | false> [false],
"schedule" : <true | false> [false],
"minute" : <0-1440> [480],
"type" : "<NONE | CHIME | FAVORITE>" ["NONE"],
"sourceId" : <1-8> [1],
"chimeId" : <1-12> [1],
"favoriteId" : <1-32> [1],
"volume" : <0 – 50> [10],
"rampVolume" : <true | false> [true],
"duration" : <0-86400> [3600 seconds],
"snoozeTime" : <0 – 30> [15 minutes],
"shuffle" : <true | false> [false],
"repeat" : <true | false> [false],
"days": {
     "once" : <true | false> [true],
     "monday" : <true | false> [false],
     "tuesday " : <true | false> [false],
     "wednesday" : <true | false> [false],
     "thursday" : <true | false> [false],
     "friday" : <true | false> [false],
     "saturday" : <true | false> [false],
     "sunday" : <true | false> [false]
}}
```

Alarm Options

The following is the format of the JSON string for alarm options. This string can be read using the RIO GET command, GET C[c].Z[z].alarm.option.

```
{ "alarmsOptions" : {
"version" : "1.0",
"options" : {
     "chimes" : [ {
          "id" : <1-12>,
          "name" : "chime name",
          "fileURL" : "http://mbx/chimes/n/n_chime_name.mp3"
          }, { … } ],
     "sources" : [ {
          "id" : <1-8>,
          "name" : "<source name>",
          "type" : "<source type>",
          "chimeSupport" : <true | false>
          }, { … } ],
     "favorites" : [ {
          "id" : <1-32>,
          "name" : "<system favorite name>",
          "mode" : "<provider mode>",
          "imgURL" : "<url>",
          "sourceType" : "<source type>"
          }, { … } ]
}}}
```

Alarm GET Commands

C[c].Z[z].alarm.status	Alarm status JSON string
C[c].Z[z].alarm.option	Alarm options JSON string
C[c].Z[z].alarm.config	Most recent JSON string used for alarm configuration

Alarm EVENT Commands

saveAlarm

The saveAlarm command is used to configure an alarm.

EVENT C[c].Z[z]!saveAlarm "<configString>"

Example:

The follow shows an example of the string with escaped characters, required when using the RIO EVENT command to configure the alarm.

EVENT C[c].Z[z]!saveAlarm
"{\"valid\":true,\"name\":\"Alarm1\",\"id\":1,\"enable\":true,\"minute\":480,\"type\":\"CHIME\",\"sourceId\":1,\"chimeId\":2,\"favoriteId\":1,\"volume\":10,\"rampVolume\":true,\"duration\":3600,\"snoozeTime\":15,\"shuffle\":false,\"repeat\":false,\"days\":{\"once\":false,\"monday\":true,\"tuesday\":true,\"wednesday\":true,\"thursday\":true,\"friday\":true,\"saturday\":false,\"sunday\":false}}"

snoozeAlarm

Alarm audio is temporarily dismissed for the configured amount of time using the snoozeAlarm event. This event operates on the active alarm id.

```
EVENT C[c].Z[z]!snoozeAlarm
```

dismissAlarm

Alarm audio is dismissed using the dismissAlarm event. This event operates on the active alarm id.

```
EVENT C[c].Z[z]!dismissAlarm
```

Alarm Status WATCH

Alarm status can be observed with a Zone WATCH

```
WATCH C[1].Z[1] ON

S

N C[1].Z[1].alarm.status="{"zoneAlarms":{"version":"1.0","alarms":[]}}"
```

Media Management

RIO includes a set of commands specifically for Media Management. This allows for content filtering, browsing and selection. These commands provide access to media contained on all Russound Media Streamer source devices.

RIO supports a Media Management model that allows a simple implementation to mimic the existing navigation model used on the ***MyRussound™*** app (iOS/Android). By implementing four screen templates, each containing a small set of fixed buttons and static text fields, it is possible to provide full Media Management of all Russound Media Streamers. These screen templates are presented in the subsequent sections in detail. Note that Media Management is referred to as 'MM' in these sections.

RIO allows for one Media Management session per IP socket connection. With C-Series and MCA-Series controllers, a single Media Management session is also available via RS232 interface.

Russound Media Management is supported only on Russound Media Streamers.

RIO MM Commands and Responses

This section presents the RIO commands and responses used to access Media Management functionality.

RIO MM Commands

MM Commands are sent by the RIO client application to Russound system that support RIO (such as the MCA-Series controllers). The first 4 RIO MM commands (MMVerbosity, MMIndex, MMMaxItems and MMFormat) are intended to configure the MM session. These commands affect the way the MM responses, or ‘notifications’, are presented. For this reason, it is best for a RIO client application to issue these commands first.

Specify Menu Notification Verbosity

This command is used to specify the amount of detail that should be provided in each MMMenuItem notification.

EVENT C[c].Z[z]!MMVerbosity [1 | 2]

Verbosity level 1 results in only the MMMenuItem.text notifications to be transmitted.

Verbosity level 2 results in all verbosity level 1 notifications as well as MMMenuItem.attr notifications.

Details of the MMMenuItem are described in subsequent sections.

Specify Menu Indexing Method

This command specifies how the MMMenuItem indices are to be interpreted.

EVENT C[c].Z[z]!MMIndex “[RELATIVE | ABSOLUTE]”

MMIndex RELATIVE interprets the index as a relative index of 1 thru 5 per pagination (pagination is provided via the MMPrevItems and MMNextItems commands, described later in this section). This is good for simple fixed line displays.

MMIndex ABSOLUTE interprets the indices as an absolute index of 1 to ‘max items’. This is good for touch screen or app based user interfaces which utilize smooth scrolling.

Specify Menu Notification Max Items

This command is used to specify the maximum number of items that should be delivered in response to an MM Menu operation (MMInit, MMSelectItem, MMPrevScreen, MMPrevItems and MMNextItems). This setting is only relevant when MMIndex is set to ABSOLUTE. A setting of MMIndex as RELATIVE always results in 5 items returned per request.

EVENT C[c].Z[z]!MMMaxItems <1 to 255>

Specify Menu Notification Format

This command specifies the format to be used for the MMMenuItem Notification messages. MMMenuItem RIO notifications always begin with the letter ‘N’ and are sent as individual messages per menu item. JSON menu item notifications are packed into a single JSON-formatted string per each pagination command. The number of items contained within a single JSON notification is specified by using the MMMaxItems command. You can expect to get better performance from the JSON format. For Javascript-based RIO/SE client applications, notification parsing will be easier as well. See **JSON Response Format** section below.

EVENT C[c].Z[z]!MMFormat “[RIO | JSON]”

JSON Response Format

If MMFormat is set to JSON the menu item notifications will be return in the following format:

Field Descriptions

totalItems - the total number of items available in this menu

numItems - the number of items in this notification list

menuItems - an array of menu items containing these fields:

id - the instance of this menu item

text - the text to be displayed for this menu item

If the MMVerbosity is set to 2, these JSON elements should be present:

isFirst: if true, this is the first menu item in the menu

isLast: if true, this is the last menu item in the menu

isMenu: if true, this menu item is a menu selection

BOT: If true, indicates that this item is the first item in a given transmission.

EOT: If true, indicates that this item is the last item in a given transmission.

value: (optional) The value of the item to be displayed (not to be confused with ‘text’)

imgURL: (optional) URL of associated album art or station art.

uri: (optional) URI associated with items that have Invokable or Hyperlink attributes.

Attributes: (optional) list of attributes (see Menu Item Attributes section).

Note: There are also attributes for isFirst, isLast, isMenu. If attributes are defined, isFirst, isLast, isMenu can be treated as deprecated.

Example:

```
{"totalItems":37,"numItems":37,"menuItems":

[{"id":1,"text":"FM","isFirst":true,"isLast":false,"isMenu":false,"BOT":true,"EOT":false,
"attributes":"Bfh"},

{"id":2,"text":"89.1 | NHPR (US News) | The Moth","imgURL":"http://cdn-radiotime-
logos.tunein.com/s28632q.png","isFirst":false,"isLast":false,"isMenu":false,"BOT":false,"
EOT":false,"attributes":"JcS"},

{"id":3,"text":"Embedded library version","value":"HEAD-v3.52.18-
g36a3fdc3","isFirst":false,"isLast":true,"isMenu":false,"BOT":false,"EOT":true,"attribute
s":"ELJsd"}

{"id":4,"text":"Open Spotify
App","uri":"spotify:/openApp","isFirst":true,"isLast":false,"isMenu":false,"BOT":true,"EO
T":false,"attributes":"BFJI"}

]}
```

Specify Info Screen Format

This command specifies how to receive INFO screen text. If set to TRUE, the text arrives in one block. If set to FALSE the text arrives in up to 4 separated text fields.

EVENT C[c].Z[z]!MMUseBlockInfo [TRUE | FALSE]

Specify Text Entry Format

This command specifies how to receive TEXT entry prompts. If set to TRUE, all text entry prompts arrive at once. If set to FALSE, text entry prompts arrive one at a time.

EVENT C[c].Z[z]!MMUseForms [TRUE | FALSE]

Navigate to a Menu Item by Index

This EVENT is used to navigate to a menu item that was received via notification. Subsequent MMPrevItems and MMNextItems will be made relative to the new menu position.

EVENT C[c].Z[z]!MMStartItem <1 to 2^32>

Note: This command is only available when MMIndex is set to ABSOLUTE.

MM Session Initialization

This event restarts MM menu state within the source device. The menu navigation is reset to the topmost (i.e., home) menu. Browsing menus and ultimately making a selection of a playable item will bring the session to the NowPlaying screen.

EVENT C[c].Z[z]!MMInit

If it is advantageous for the session to respond to Key Events for transport controls (e.g. Play, Pause, Next, etc) *before* a playable selection has been made, the session may be moved directly to the NowPlaying screen by using the MMClose command (see below).

MM Session Termination

This event ends the MM session and returns the RIO session to the NowPlaying screen.

EVENT C[c].Z[z]!MMClose

After issuing this RIO MM command, a source-specific NowPlaying screen notification will be sent to the RIO client. The NowPlaying screen notification is described in the **Screen Change Notifications** section below.

Moving the MM session to the NowPlaying screen enables Key Events for transport controls (e.g. Play, Pause, Next, etc) to be processed.

Menu Item Selection

This EVENT is used to select from a list of items presented by the previous set of text field notifications (see **Menu Item Notifications** below). The number of valid selections is dependent on the number of MMItems that are received from the RIO source device and the value of MMIndex (RELATIVE or ABSOLUTE), depending on the current screen being displayed.

If MMIndex is set to RELATIVE,

EVENT C[c].Z[z]!MMSelectItem [1 to 6]

If MMIndex is set to ABSOLUTE,

EVENT C[c].Z[z]!MMSelectItem [1 to max items]

Setting Optional Value

MMSelectItem can also be used to set an optional value when selected.

EVENT C[c].Z[z]!MMSelectItem [1 to 6] [<value>]

If MMIndex is set to ABSOLUTE,

EVENT C[c].Z[z]!MMSelectItem [1 to max items] [<value>]

Setting values is only valid when used with menu items that have the following attributes:

Editable

EditTypeString

EditTypeIPAddress

EditTypeNumber

EditTypeSlider

EditTypeRadioBox

EditTypeBool

Item Context Menu Selection

This EVENT is used to select the context menu of an item in a list of items presented by the previous set of text field notifications (see **Menu Item Notifications** below). The number of valid selections is dependent on the number of MMItems that are received from the RIO source device and the value of MMIndex (RELATIVE or ABSOLUTE), depending on the current screen being displayed. This command should only be used if the menu item attribute includes the HasContextMenu attribute.

If MMIndex is set to RELATIVE,

EVENT C[c].Z[z]! MMItemContextMenu [1 to 6]

If MMIndex is set to ABSOLUTE,

EVENT C[c].Z[z]! MMItemContextMenu [1 to max items]

Screen Data Confirmation

This EVENT is used as confirmation for the current screen data and advances the Media Management session to the next navigable screen. The Info and TextEntry screens for the Russound Media Streamer make use of this MM EVENT. These screens are described in more details in subsequent sections.

EVENT C[c].Z[z]!MMSelectOk

Item Navigation

These EVENTs are used to request the next and previous set of items from the current list.

EVENT C[c].Z[z]!MMNextItems

EVENT C[c].Z[z]!MMPrevItems

Screen Navigation

This EVENT requests that the previous screen be selected. This may result in the system sending a screen change notification (see Screen Change Notifications below) as well as text field change notifications (see Menu Item Notifications below).

EVENT C[c].Z[z]!MMPrevScreen

Text Edit Navigation

These EVENTs allow editing of alphanumeric fields, where the text search feature is available.

EVENT C[c].Z[z]!MMCursorNext

EVENT C[c].Z[z]!MMCursorPrev

EVENT C[c].Z[z]!MMLetterUp

EVENT C[c].Z[z]!MMLetterDown

The Context Menu

Once a selection has been made, the user interface will be brought to the Now Playing screen which displays information about the current selection. Certain services, such as Pandora, support retrieval of additional context specific information. A Context Menu is invoked by using:

EVENT C[c].Z[z]! MMContextMenu

This event invokes a new set of menu items containing choices relative to the particular service. Navigation of the context menus is done using the same events as are used for the other selection menus.

Likes and Dislikes

Some services support the ability for the user to express preferences about the current selection. This is done with an event:

EVENT C[c].Z[z]!MMRate high | low

Where *high* is LIKE and *low* is DISLIKE.

When songs change, rating information is sent as notifications that appear when the source is being watched. They will appear as:

N S[2].rating="UNKNOWN"

This indicates no rating information for that song has been previously provided, and is the default behavior. If a preference has been provided, values of “LIKE” or “DISLIKE” will be shown.

Text Field Entry

This EVENT allow direct entry of a text field, where the text search feature is available. This approach should be used by advanced UI devices capable of presenting an ASCII keyboard to the user for text entry. The TextEntry screen for the Russound Media Streamer makes use of this MM EVENT.

```
EVENT C[c].Z[z]!MMTextField "<text string>"
```

Select Menu Option

Some menu items have attributes which indicate that the menu item is the top level of a music service. (see the Menu Item Attributes section) If these menu items do not also include the ‘M’ attribute (sub-menu item), this means that the last played channel or song for that service will be played upon selection. If the player is already playing content from that service, or if no last played content is available for that service, the ‘M’ attribute will be set and selection will always result in browsing into submenus.

The MMSelectOption command can be used to override this behavior. The command should be issued immediately before sending the MMSelectItem command. MMSelectOption only has effect on menu items that have service type attributes set.

Possible values:

“default” – Default behavior as explained above.

“norestore” – This event option allows you to override the content restore and enter the sub-menu for the selected music service.

```
EVENT C[c].Z[z]!MMSelectOption norestore
```

Keep Alive

Call this to keep media management from timing out (1 minute) due to lack of user input.

EVENT C[c].Z[z]!MMKeepAlive

RIO MM Responses

Screen Change Notifications

This section presents the RIO messages ***received by*** the Third Party Device to indicate that a new screen template must be displayed. This RIO message is typically followed by a series of RIO messages containing text, used to populate the text fields within the screen to be displayed. For more information on these text fields messages, see the ***Menu Item Notifications*** section.

N S[s].MMScreen=”<Screen ID>”

where Screen ID is defined as:

Russound Media Streamer Screen Identifiers

Screen ID	Screen Name
SourceMenuScreen	Menu
SourceInfoScreen	Info
SourceNowPlayingScreen	Now Playing
SourceTextEntryScreen	Text Entry

Screen Title Notifications

All screens can expect notifications of changes to the screen title. These will appear as notification messages in this format:

Screen Title Text

N S[s].MMTitle.text=”<title string>”

Screen Title Attributes

If MMVerbosity is set to ‘2’, title attribute notifications will also be transmitted. Valid title attributes include HomeMenu, (“H”), UserLogin (“U”), any one of the streaming service attributes. Combination of UserLogin and a streaming service attribute identify the screen as the login screen for a specific service (i.e. SiriusXM).

N S[s].MMTitle.attr=<text string>

Menu Item Notifications

Menu screens can expect notifications of changes to Menu Item text. These will appear as notification messages in this format:

N S[s].MMMenuItem[1 to 2^32].text=”<text string>”

Menu Item Text

N S[s].MMMenuItem[1 to 2^32].text=”<text string>”

Menu Item Attributes

If MMVerbosity is set to ‘2’, menu item attribute notifications will also be transmitted:

N S[s].MMMenuItem[1 to 2^32].attr=”<text string>”

The attr string is a series of letters that provide further details about the menu item. Note that a text string may have multiple attributes. The following table lists the display attributes used in MenuItem notifications.

Attribute	Attribute Key	Definition	New For MBX
BOT	“B“	Beginning Of Transmission. Marker for first in a list of indeterminate length.	
EOT	“E“	End of Transmission. Marker for last in a list of indeterminate length.	
LastMenuItem	“L“	Indicates this is the last item in a menu. Useful for hiding/showing the Page Down control.	
SubMenuItem	“M“	Sub-menu item.	
FirstMenuItem	“F“	Indicates this is the first item in a menu. Useful for 'hiding' the Page Up Control	
Hidden	“X“	Menu item is hidden	
CurrentMenuItem	“C“	Indicates the currently selected menu item. This should only be asserted to indicate the item 'uniquely' on the UI.	
ActiveMenuItem	“A“	Indicates the menu item is in an active state. Usually to indicate a setting state.	
PlayingMenuItem	“P“	Indicates the current menu item is playing	
HomeMenu	“H“	Indicates a home or top menu. Typically used when transmitting the menu title text.	
UserLogin	“U”	Indicates that the screen is a login screen. This is normally paired with one of the service type attributes below to identify a specific service login screen. Typically used when transmitting the menu title text	
JSON	“J“	Display message in JSON format	
Subtext	“S“	Display message has subtext delimited by “ | “.	*
Invokable	“I“	Item is invokable and should be displayed as a button	*
Header	“h“	Item is a non actionable menu header.	*

HasContextMenu	“c“	Item has a context menu	*
Hyperlink	“w“	Item has a hyperlink	*
Editable	“e“	Item is editable	*
EditTypeString	“s“	Edit type is string	*
EditTypeIPAddress	“i“	Edit type is IP address	*
EditTypeNumber	“n“	Edit type is number	*
EditTypeSlider	“l“	Edit type is slider	*
EditTypeExecute	“x“	Edit type is executable (typically shown as button or link text)	*
EditTypeRadioBox	“r“	Edit type is radio box control	*
EditTypeBool	“b“	Edit type is bool (typically shown as switch or check box)	*
Disabled	“d“	Item is disabled	*
Pandora	“!”	Service topmost menu item or service specific screen title.	
TuneIn	“@“	Service topmost menu item or service specific screen title	
SiriusXM	“#“	Service topmost menu item or service specific screen title	
vTuner	“$“	Service topmost menu item or service specific screen title	
DLNA	“%“	Service topmost menu item or service specific screen title	
USB	“^“	Service topmost menu item or service specific screen title	
Spotify	“&“	Service topmost menu item or service specific screen title	
Party	“*“	Service topmost menu item or service specific screen title	
Tidal	“(“	Service topmost menu item or service specific screen title	*

Napster	“)“	Service topmost menu item or service specific screen title	*
Deezer	“-“	Service topmost menu item or service specific screen title	*
Bluetooth	“_“	Service topmost menu item or service specific screen title	*
SPDIF	“+“	Service topmost menu item or service specific screen title	*
LineIn	“[“	Service topmost menu item or service specific screen title	*
Chromecast	“]“	Service topmost menu item or service specific screen title	*
AirableRadio	“|“	Service topmost menu item or service specific screen title	*

In response to the pagination commands, ***MMPrevItems*** and ***MMNextItems***, these notifications are transmitted:

MMMenu.totalItems

This notification indicates the total number of items present in the current menu

N S[s].MMMenu.totalItems=””

MMMenu.numItems

This notification indicates the number of items present in the most recent pagination request:

N S[s].MMMenu.numItems=””

Info Screen Notifications

Info Screen Buttons

Info screens may need to display 0 or more buttons to indicate what actions are required for the screen. Each notification indicates the type of the button and the string to be displayed to the user for that button:

```
N S[2].MMBtnOK.text=”<text string>”
N S[2].MMBtnBack.text=”<text string>”
```

If no button notifications are present for a screen, no buttons should be displayed.

Info Screen Text

Info screens display 4 lines of text, which appear as text with attributes:

```
N S[2].attr="BF"
N S[2].MMInfo[1].text="If you want to logout of"
N S[2].MMInfo[2].text="SiriusXM, press OK. If not,"
N S[2].MMInfo[3].text="press the Back/Prev button."
N S[2].attr="EL"
N S[2].MMInfo[4].text=""
```

If there is not enough text to fill 4 lines, empty lines will be provided as padding. The first and last lines are indicated by attributes; if more than 4 lines are present, the text may be scrolled with KeyRelease PageUp and KeyRelease PageDown events.

Info Screen Changes

The Info screen can expect notifications of changes to the info fields. These will appear as notification messages in this format:

N S[s].MMInfo[1-4].text=”<info string>”

Now Playing Notifications

Source and zone notifications such as metadata and volume drive updates of the Now Playing screen. Source and zone notifications may arrive prior to a screen change notification, so values should be cached as they are received.

Like and Dislike

When songs change, rating information is sent as notifications that appear when the source is being watched. They will appear as one of the following:

N S[2].rating="UNKNOWN"

N S[2].rating="LIKE"

N S[2].rating="DISLIKE"

Text Entry Screen Notifications

The Text Entry screen can expect notifications of changes to the text entry field (where user text is entered) and help text (describing what type of information the user should enter into the text entry field. These will appear as notification messages in this format:

N S[s].MMTextField.text=”<text string>”

N S[s].MMHelp.text=”<text string>”

Forms

It is possible to have multiple prompts and multiple fields on a single Text Entry screen for user input. This is referred to as “using forms’; using forms is enabled with an event:

```
EVENT C[1].Z[1]!MMUseForms true
```

Once this is done, if the Russound Media Streamer has text entry screens with multiple input lines (e.g. a screen with ‘username’ and ‘password’) the following notifications will be seen:

```
N S[2].MMScreen="SourceTextEntryScreen"

N S[2].MMTitle.text="SiriusXM"

N S[2].MMHelp.text="Enter your Username"

N S[2].MMForm1Field.text="Russound40"

N S[2].MMHelp2.text="Enter your Password"

N S[2].MMForm2Field.text="siriusxm"
```

Note that the MMHelp field is augmented by field ‘MMHelp2’. The MMTextField is replaced by ‘MMForm1Field’ and ‘MMForm2Field’.

When MMUseForms is true, you may set the text fields in any order (and as often as you like) using the MMFormEntry event:

```
EVENT C[1].Z[1]!MMFormEntry "user" 1

EVENT C[1].Z[1]!MMFormEntry "password" 2
```

Where the first argument is the string to be used and the second argument is the number of the MMForm<n>Field to which the string is to be applied.

Once all Form Fields have been set, use

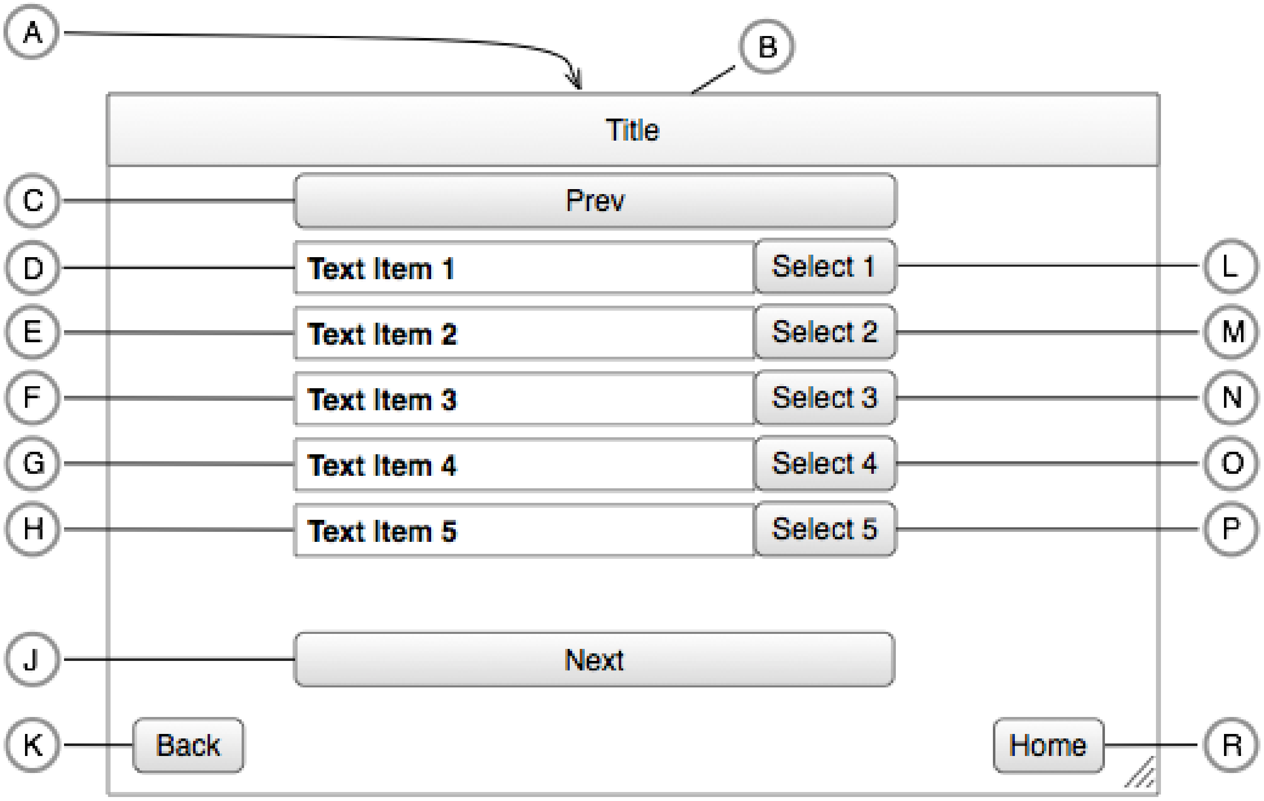
```
EVENT C[1].Z[1]!MMSelectOk
```

to submit the form and move on to subsequent screens

Russound Media Streamer Example Screens

Menu Screen

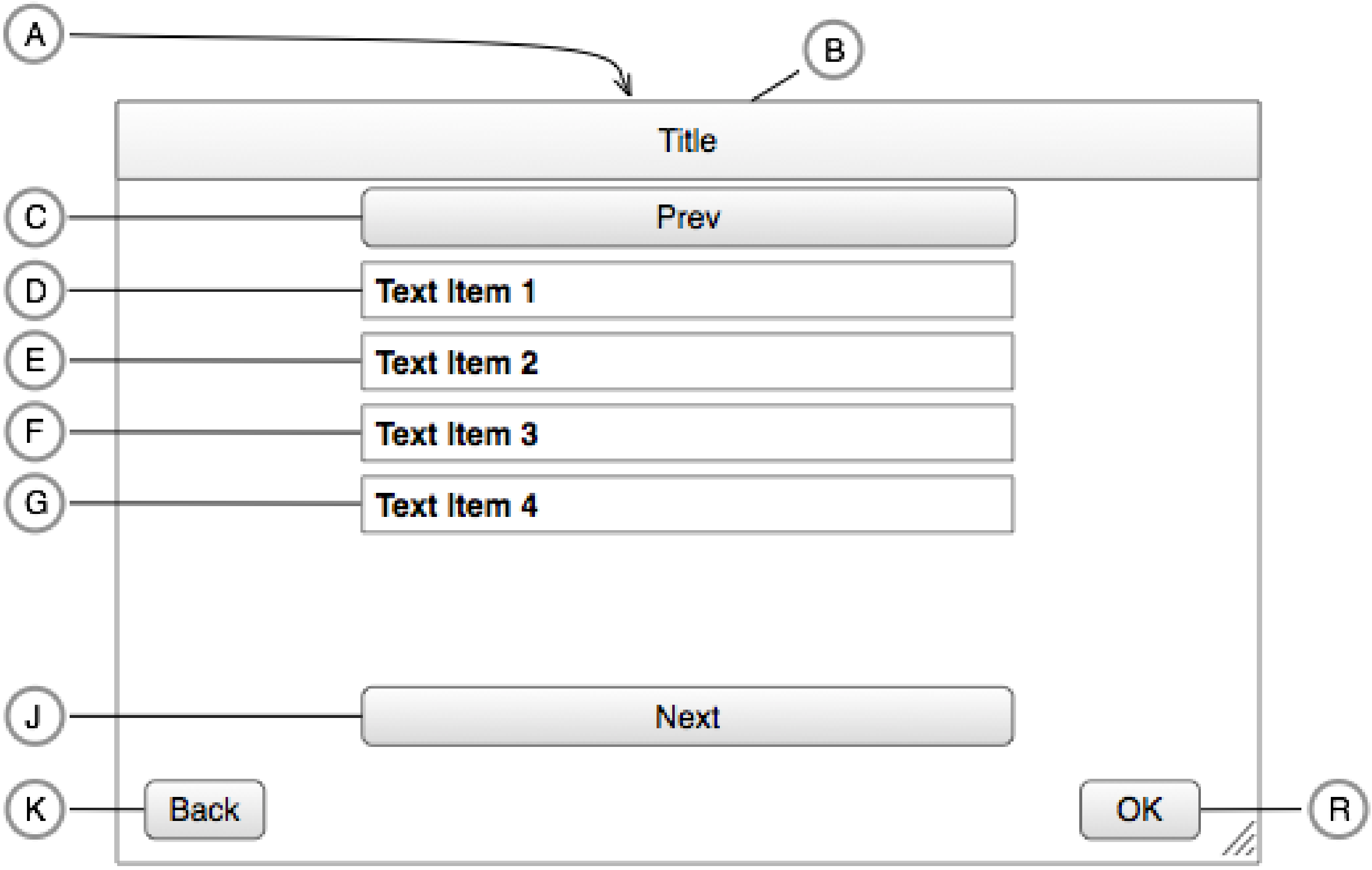
The example below uses MMIndex set to RELATIVE.



Menu Screen Elements

Screen Designator	RIO Command/Response
A	N S[s].MMScreen=”SourceMenuScreen”
B	N S[s].MMTitle.text=”<title string>”
C	EVENT C[c].Z[z]! KeyRelease PageUp
D	N S[s].MMMenuItem[1].text=”<text string>”
E	N S[s].MMMenuItem[2].text=”<text string>”
F	N S[s].MMMenuItem[3].text=”<text string>”
G	N S[s].MMMenuItem[4].text=”<text string>”
H	N S[s].MMMenuItem[5].text=”<text string>”
I	N/A
J	EVENT C[c].Z[z]!KeyRelease PageDown
K	EVENT C[c].Z[z]!MMPrevScreen
L	EVENT C[c].Z[z]!MMSelectItem 1
M	EVENT C[c].Z[z]!MMSelectItem 2
N	EVENT C[c].Z[z]!MMSelectItem 3
O	EVENT C[c].Z[z]!MMSelectItem 4
P	EVENT C[c].Z[z]!MMSelectItem 5
Q	N/A
R	EVENT C[c].Z[z]!MMInit
S	N/A
T	N/A
U	N/A
V	N/A

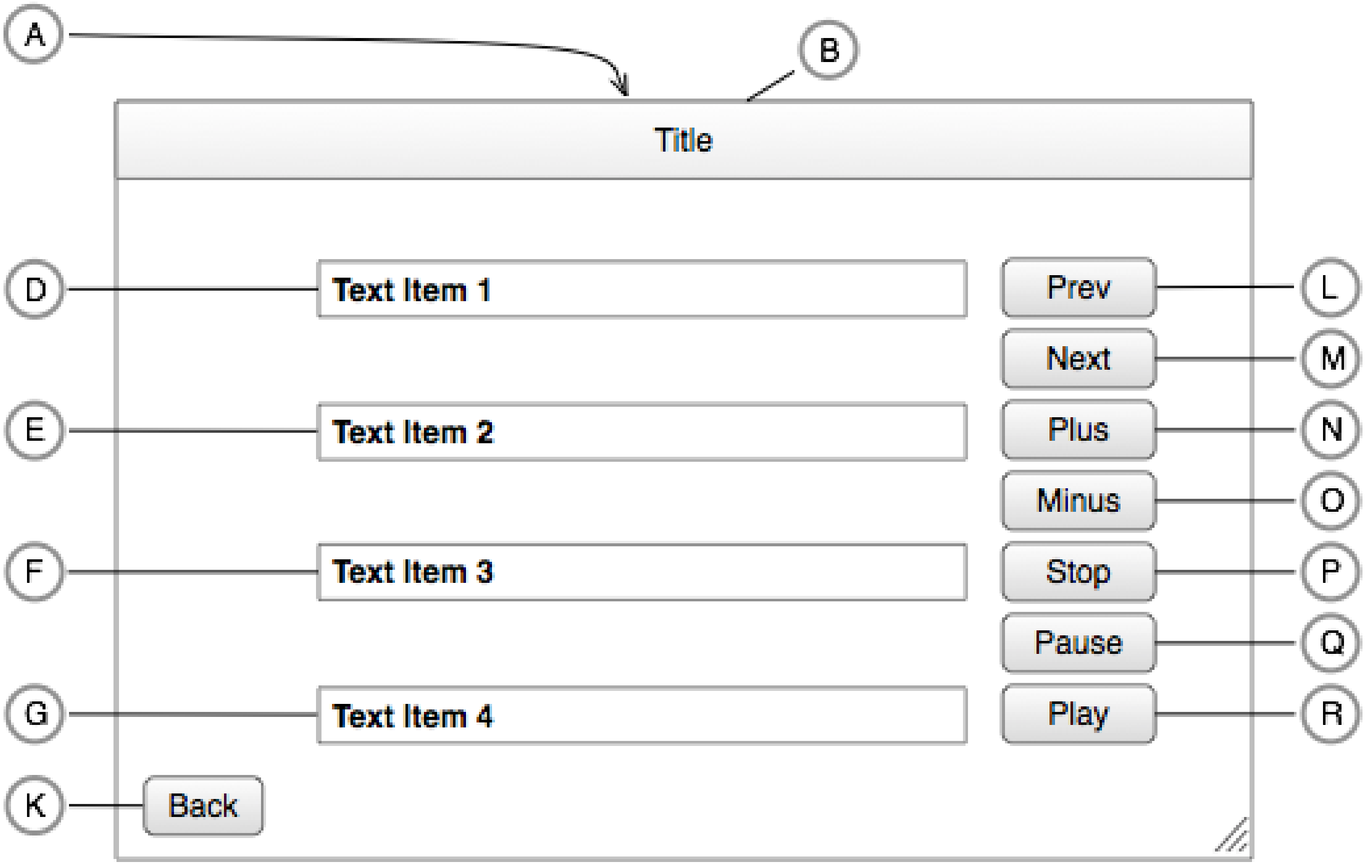
MM Info Screen



Info Screen Elements

Screen Designator	RIO Command/Response
A	N S[s].MMScreen=”SourceInfoScreen”
B	N S[s].MMTitle.text=”<title string>”
C	EVENT C[c].Z[z]!KeyRelease PageUp
D	N S[s].MMMenuItem[1].text=”<text string>”
E	N S[s].MMMenuItem[2].text=”<text string>”
F	N S[s].MMMenuItem[3].text=”<text string>”
G	N S[s].MMMenuItem[4].text=”<text string>”
H	N/A
I	N/A
J	EVENT C[c].Z[z]!KeyRelease PageDown
K	EVENT C[c].Z[z]!MMPrevScreen
L	N/A
M	N/A
N	N/A
O	N/A
P	N/A
Q	N/A
R	EVENT C[c].Z[z]!MMSelectOk
S	N/A
T	N/A
U	N/A
V	N/A

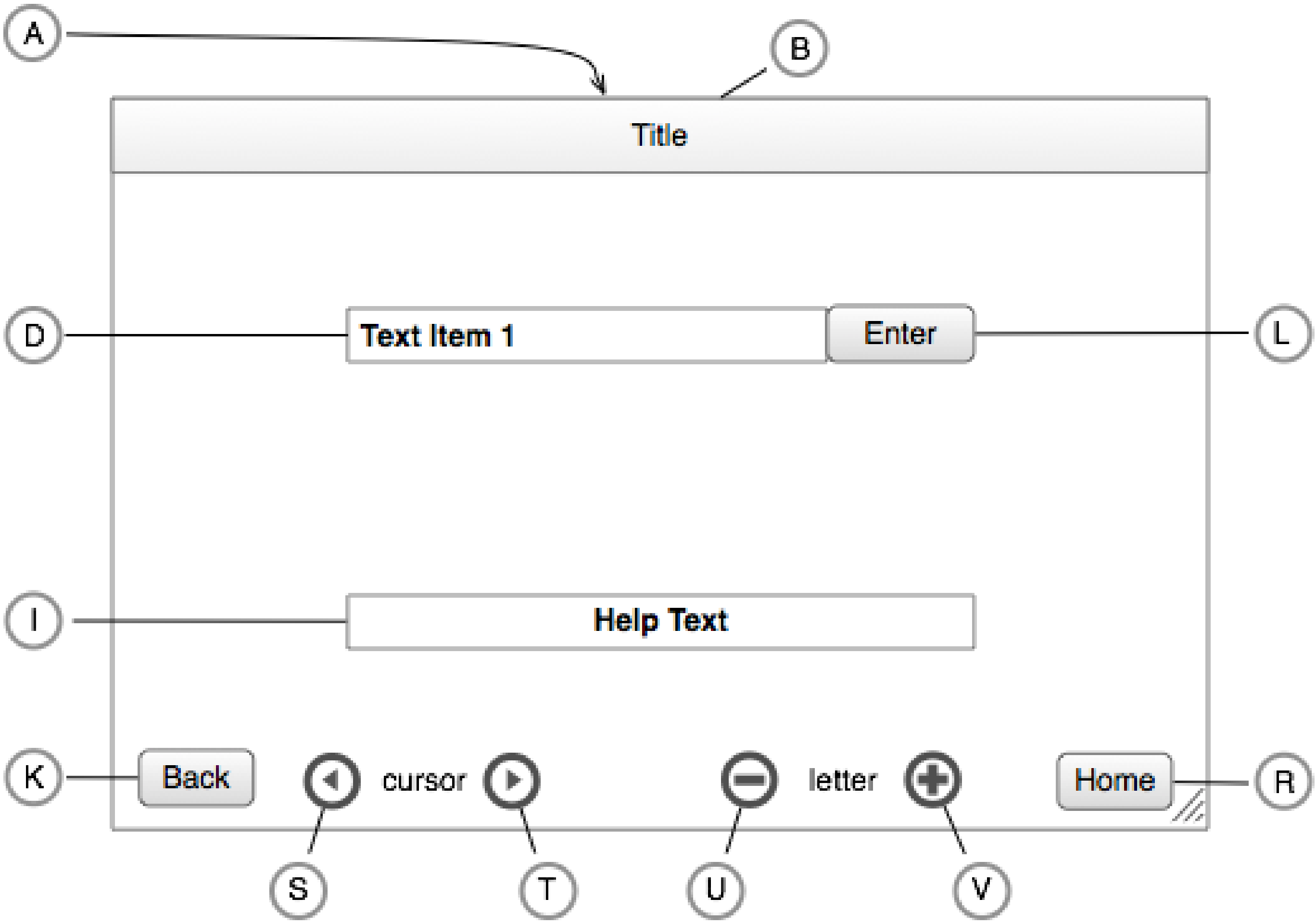
Now Playing Screen



Now Playing Screen Elements

Screen Designator	RIO Command/Response
A	N S[s].MMScreen=”SourceNowPlayingScreen”
B	N S[s].MMTitle.text=”<title string>”
C	N/A
D	N S[s].playlistName=”<value>” / N S[s].channelName=”<value>”
E	N S[s].artistName=”<value>”
F	N S[s].albumName=”<value>”
G	N S[s].songName=”<value>”
H	N/A
I	N/A
J	N/A
K	EVENT C[c].Z[z]!MMPrevScreen
L	EVENT C[c].Z[z]!KeyRelease Previous
M	EVENT C[c].Z[z]!KeyRelease Next
N	EVENT C[c].Z[z]!KeyRelease ChannelUp
O	EVENT C[c].Z[z]!KeyRelease ChannelDown
P	EVENT C[c].Z[z]!KeyRelease Stop
Q	EVENT C[c].Z[z]!KeyRelease Pause
R	EVENT C[c].Z[z]!KeyRelease Play
S	N/A
T	N/A
U	N/A
V	N/A

Text Entry Screen



Text Entry Screen Elements

Screen Designator	RIO Command/Response
A	N S[s].MMScreen=”SourceTextEntryScreen”
B	N S[s].MMTitle.text=”<title string>”
C	N/A
D	N S[s].MMTextField.text=”<text string>”
E	N/A
F	N/A
G	N/A
H	N/A
I	N S[s].MMHelp.text=”<text string>”
J	N/A
K	EVENT C[c].Z[z]!MMPrevScreen
L	EVENT C[c].Z[z]!MMTextField "<text string>"
M	N/A
N	N/A
O	N/A
P	N/A
Q	N/A
R	N/A
S	EVENT C[c].Z[z]!MMCursorPrev
T	EVENT C[c].Z[z]!MMCursorNext
U	EVENT C[c].Z[z]!MMLetterUp
V	EVENT C[c].Z[z]!MMLetterDown

Summary of Events

SelectSource

ZoneOn

ZoneOff

ZoneMuteOn

ZoneMuteOff

AllOn

AllOff

KeyPress

KeyRelease

KeyHold

KeyCode

PartyMode

Shuffle

Repeat

DoNotDisturb

SaveSystemFavorite

SaveZoneFavorite

RestoreSystemFavorite

RestoreZoneFavorite

DeleteSystemFavorite

DeleteZoneFavorite

SavePreset

RestorePreset

DeletePreset

Summary of Media Management Events

MMInit

MMClose

MMSelectItem

MMNextItems

MMPrevItems

MMPrevScreen

MMCursorNext

MMCursorPrev

MMLetterUp

MMLetterDown

MMMaxItems

MMStartItem

MMItemContextMenu

MMSelectOk

MMTextField

MMVerbosity

MMFormat

MMIndex

MMRate

MMUseForms

MMUseBlockInfo

MMFormEntry

MMSelectOption

MMContextMenu

MMKeepAlive

DEPRECATED

MMStartChar – use MMStartItem

Summary of Media Management INFO Screen Notifications

The following INFO screen notifications pertain to DMS-3.1 and XStream-Series.

Pandora Acct Inactive Info

Title: "Inactive Account"

Text: "Sorry, this account is no longer active. Please contact Pandora listener support at pandora-support@pandora.com."

BtnOK: "OK"

Pandora Confirm Logout

Title: "Sign Out"

Text: "Are you sure that you want to Sign Out of Pandora?"

BtnOK: "OK"

BtnBack: "Cancel"

Pandora Empty Search Info

Title: "Error"

Text: “You entered a blank string. Please try again."

BtnOK: "OK"

Pandora Invalid Country

Text: "Sorry, Pandora is not available in this country"

BtnOK: "OK"

Pandora Invalid Login

Title: "Login Failed"

ONE OF THE FOLLOWING:
Text: "User must sign in to re-authenticate the device."
Text: "Username or password is invalid."
Text: "Sorry, this account is no longer active. Please contact Pandora listener support at pandora-support@pandora.com."
Text: "User is not authorized."
Text: "Pandora is performing system maintenance. Please try again later."
Text: "Complimentary period for this device has expired."
Text: "Unable to communicate with Server. Please check your network settings and try again."
Text: "This device is not registered with Pandora."
Text: "Device partner is not authorized for this action."
Text: "Username is invalid."
Text: "Password is invalid."

Text: "Username provided is already is use."
Text: "This device is already associated to an account."
Text: "Email address is inavlid."
Text: "This device model is invalid."
Text: "User account is not found."
Text: "Internal server error."

BtnOK: "OK"

Pandora Invalid Logout

Title: "Logout Failed"

Text: "Pandora is performing system maintenance. Please try again later"

BtnOK: "OK"

Pandora Option Delete Station

Title: "Delete Station"

Text: "Are you sure that you want to delete [currentStation]?"

BtnOK: "Delete"

BtnBack: "Cancel"

Sirius Confirm Logout

Title: "Logout"

Text: “Are you sure that you want to logout of SiriusXM?"

BtnOK: "OK"

BtnBack: "Cancel"

Sirius Invalid Login

Title: "Login Failed"

ONE OF THE FOLLOWING:
Text: "General Error (SiriusXM). Please contact your Russound Installer."
Text: "Unknown Error (SiriusXM). Please contact your Russound Installer."
Text: "Invalid Device (SiriusXM). Please contact your Russound Installer."
Text: "Service Not Found (SiriusXM). Please contact your Russound Installer."
Text: "Upgrade Required (SiriusXM). Please contact your Russound Installer."
Text: "Missing Parameter Error (SiriusXM). Please contact your Russound Installer."
Text: "Service Unavailable (SiriusXM). Please contact your Russound Installer."
Text: "Firmware Update Required (SiriusXM). Please contact your Russound Installer."
Text: "Invalid Device ID (SiriusXM). Please contact your Russound Installer."
Text: "Device Temporarily Unavailable (SiriusXM). Please try again later."
Text: "Device Disabled (SiriusXM). Please contact your Russound Installer."

Text: "Authentication Required (SiriusXM). Please contact your Russound Installer."
Text: "Authentication password is incorrect (SiriusXM). Please contact your Russound Installer."
Text: "Authentication Failed (SiriusXM). Please contact your Russound Installer."
Text: "Authentication session has expired (SiriusXM). Please contact your Russound Installer."
Text: "Authentication (SiriusXM): Invalid Sequence Error. Please contact your Russound Installer."
Text: "Authentication (SiriusXM): Bad Session Error. If this persists, please contact Russound Technical Support."
Text: "Password expired (SiriusXM). Please contact your Russound Installer."
Text: "Invalid Client Certificate (SiriusXM). Please contact your Russound Installer."
Text: "Services Login Required. Please contact your Russound Installer."
Text: "The Username and Password are not valid. Please try again or contact SiriusXM to subscribe."
Text: "External Authorization Required (SiriusXM). Please contact your Russound Installer."
Text: "Channel Lineup Out Of Date (SiriusXM). Please contact your Russound Installer."
Text: "Insufficient Authentication Level Error (SiriusXM). Please contact your Russound Installer."
Text: "Radio Authentication Failure (SiriusXM). Please contact your Russound Installer."
Text: "Free SiriusXM trial expired. Please contact SiriusXM to subscribe."
Text: "Unable to Login. Please contact SiriusXM support."
Text: "Unable to Login. Please contact SiriusXM support."
Text: "Unable to Login (SiriusXM). You have exceeded the maximum daily logins."
Text: "Unknown error. Please contact your Russound Installer."

BtnOK: "OK"

TuneIn Acct Association Info

Title: “Account Association”

```
if (Associated)
    Text: "Your account is associated with [accountName]”
        "To disassociate this device from your account, visit http://tunein.com/devices"
else
    Text: "Your Association Code is [associationCode]. To associate this device with your account, visit
http://tunein.com/devices"
        "To create an account, visit http://tunein.com"
```

BtnOK: "OK"

TuneIn Search Info

Title: "TuneIn Search Info"

ONE OF THE FOLLOWING:

Text: *Search Results*

Text: "No search results for [search string]. Please try again."

BtnOK: "OK"

Party Leave

Title: "Leave Party?"

Text: " Would you like to leave this party?"

BtnOK: "Leave"

BtnBack: "Cancel"

Party Share

Title: "Start Sharing?"

Text: "Would you like to start sharing?"
"You can join this party from another room by selecting Party->Join Party."

BtnOK: "Share"

BtnBack: "Cancel"

Party Share AirPlay

Title: "Could not share."

Text: "I'm sorry. Due to licensing restrictions, it is not possible to share AirPlay content."
"To start sharing, please select a different content type."

BtnOK: "OK"

Party Share Full

Title: "Could not share."

Text: "I'm sorry, there are no party sessions available."
"You can create up to two parties, with up to six rooms participating in each party at any given time."
"To create a new party, stop sharing from one of the other rooms by selecting Party->Stop Sharing."

BtnOK: "OK"

Party Share Unsupported Format

Title: "Could not share."

```
if (Spotify)

    Text: "I'm sorry, Spotify content cannot be shared."
          "If you would like to start a new party session, please select different content."

else

    Text: "I'm sorry, High Definition audio cannot be shared."
          "If you would like to start a new party session, please select different content."
```

BtnOK: "OK"

Party Stop

Title: "Could not share."

Text: "Would you like to stop sharing?"

BtnOK: "Stop"

BtnBack: "Cancel"

AirPlay In Use

Title: "AirPlay In Use"

Text: "To make another media selection, please discontinue AirPlay streaming."

BtnOK: "OK"

Pandora Option Info

ONE OF THE FOLLOWING:
Text: "Station deleted, please select a new station."
Text: "Pandora will play more music like this."
Text: "This track won't play again on this station."
Text: "Station Created."
Text: "Station Saved."
Text: "This option is not currently available."

BtnOK: "OK"

Server Error

Title: "Error"

Text: *Error Message*

BtnOK: "OK"

RIO Key String

The key string is formatted as a dot-separated ('.') series of strings that refer to a hierarchical set of branches, tables and leafs.

Key strings are case insensitive.

Branches, represented as a capitalized string, serve to organize the system parameters by category. A key string can contain multiple branches.

For example, in the WATCH SYSTEM command response,

```
N System.status=”<value>”
```

'System' is a branch string.

Tables, represented by a capitalized string followed by a bracketed 1-based number, allow for instances of items such as controllers, zones, source, etc to be referenced by index. A key string can contain multiple tables.

For example, in the WATCH Source command response,

```
N S[1].type=”<value>”
```

'S[s]' is a table.

Leafs, represented by a lowercase 'camel' string and always the last string in the key string, refer to a specific system parameter. A key string has only one leaf string.

For example, in the WATCH Zone command response,

```
N C[1].Z[1].bass=”10”
```

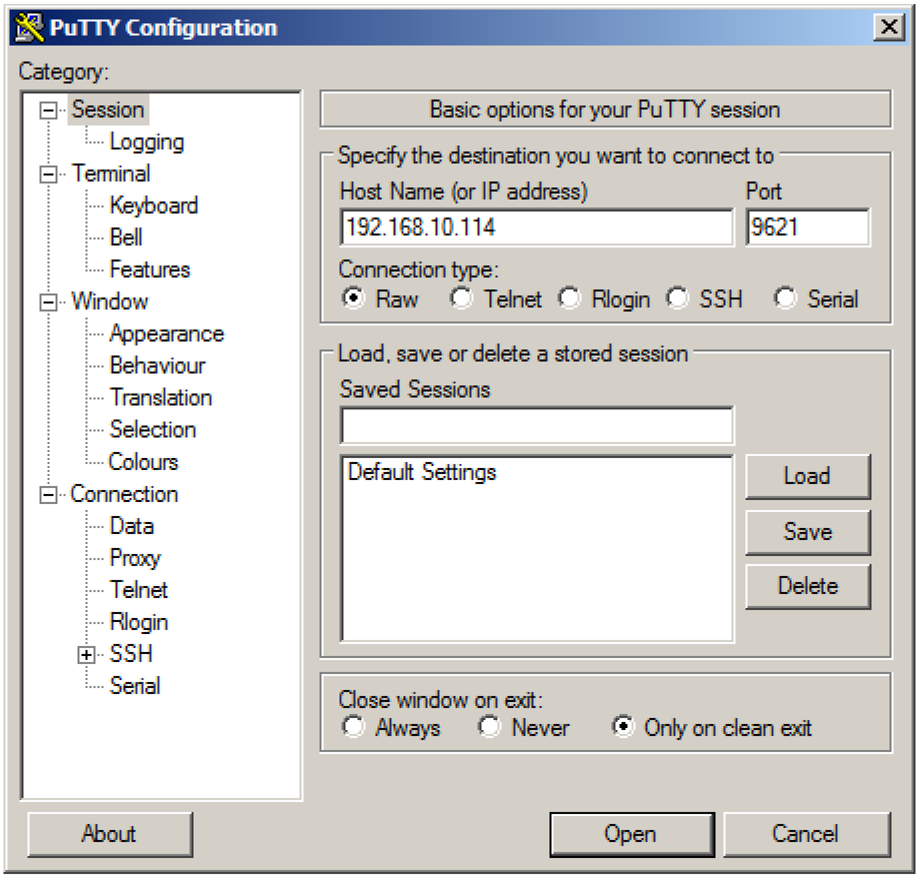
'bass' is a leaf string.

Using PuTTY as a RIO Client

PuTTY is a popular free implementation of Telnet and SSH for Win32 and Unix platforms. It is possible to use PuTTY as a RIO client for testing purposes, provided it is configured correctly.

Available here:

- Enter hostname or IP address
- Set connection type to Raw
- Set Port to 9621
- Select Open



PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): 192.168.10.114

Port: 9621

Connection type: ● Raw ○ Telnet ○ Rlogin ○ SSH ○ Serial

Load, save or delete a stored session

Saved Sessions: Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit

Table of Numeric Key Codes

Key Code	Val	Key Code	Val	Key Code	Val
1	1	Menu Down	34	Dim	67
2	2	Menu Left	35	Close	68
3	3	Menu Right	36	Open	69
4	4	Select	37	Stop 2	70
5	5	Exit	38	AM FM	71
6	6	Display	39	Cue	72
7	7	Guide	40	Disc Up	73
8	8	Page Up	41	Disc Down	74
9	9	Page Down	42	Info	75
0	10	Disc	43	External Src	76
Volume Up	11	Plus 10	44	Live/Intro	77
Volume Down	12	Open Close	45	Setup Menu	78
Mute	13	Random	46	Back	79
Chan Up	14	Track Fwd	47	Fav Channel	80
Chan Down	15	Track Rev	48	Display Fmt	81
Power	16	Sur On Off	49	SAP	82
Enter	17	Sur Mode	50	Slow	83
Prev Chan	18	Sur Up	51	PIP On	84
TV Video	19	Sur Down	52	PIP Off	85
TV VCR	20	PIP	53	PIP Freeze	86
A B	21	PIP Move	54	PIP Input	87
TV DVD	22	PIP Swap	55	PIP Chan Up	88
TV LD	23	Program	56	PIP Chan Dn	89
Input	24	Sleep	57	Input 1	90
TV DSS	25	On	58	Input 2	91
Play	26	Off	59	Input 3	92
Stop	27	11 / Red Button	60	Input 4	93

Search Fwd	28	12 / Yellow Button	61	Input 5	94
Search Rev	29	13 / Green Button	62	Input 6	95
Pause	30	14 / Blue Button	63	Input 7	96
Record	31	15 / DVR List	64	Input 8	97
Menu	32	16 / CC	65	Input 9	98
Menu Up	33	Bright	66	Input 10	99