ANSI E1.37-1 – 2012 (R2017)

Additional Message Sets for ANSI E1.20 (RDM) – Part 1, Dimmer Message Sets

CP/2009-1019r5

This edition of ANSI E1.37-1 is a reaffirmation of ANSI E1.37-1 – 2002. This reaffirmation was approved by the American National Standards Institute on 18 May 2017.

[blank page]

NOTICE and DISCLAIMER

ESTA does not approve, inspect, or certify any installations, procedures, equipment or materials for compliance with codes, recommended practices or standards. Compliance with a ESTA standard or an American National Standard developed by ESTA is the sole and exclusive responsibility of the manufacturer or provider and is entirely within their control and discretion. Any markings, identification or other claims of compliance do not constitute certification or approval of any type or nature whatsoever by ESTA.

ESTA neither guarantees nor warrants the accuracy or completeness of any information published herein and disclaims liability for any personal injury, property or other damage or injury of any nature whatsoever, whether special, indirect, consequential or compensatory, directly or indirectly resulting from the publication, use of, or reliance on this document. In issuing and distributing this document.

In issuing this document, ESTA does not either (a) undertake to render professional or other services for or on behalf of any person or entity, or (b) undertake any duty to any person or entity with respect to this document or its contents. Anyone using this document should rely on his or her own independent judgment or, as appropriate, seek the advice of a competent professional in determining the exercise of reasonable care in any given circumstance.

Published by:

The Entertainment Services and Technology Association
630 Ninth Avenue, Suite 609
New York, NY 10036
USA
Phone: 1-212-244-1505
Fax: 1-212-244-1502
Email: standards@esta.org

The ESTA Technical Standards Program

The ESTA Technical Standards Program was created to serve the ESTA membership and the entertainment industry in technical standards related matters. The goal of the Program is to take a leading role regarding technology within the entertainment industry by creating recommended practices and standards, monitoring standards issues around the world on behalf of our members, and improving communications and safety within the industry. ESTA works closely with the technical standards efforts of other organizations within our industry and represents the interests of ESTA members to ANSI, UL, and the NFPA. The Technical Standards Program is accredited by the American National Standards Institute.

The Technical Standards Council (TSC) was established to oversee and coordinate the Technical Standards Program. Made up of individuals experienced in standards-making work from throughout our industry, the Council approves all projects undertaken and assigns them to the appropriate working group. The Technical Standards Council employs a Technical Standards Manager and Assistant to coordinate the work of the Council and its working groups as well as maintain a “Standards Watch” on behalf of members. Working groups include: Control Protocols, Electrical Power, Event Safety, Floors, Fog and Smoke, Followspot Position, Photometrics, Rigging, and Stage Lifts.

ESTA encourages active participation in the Technical Standards Program. There are several ways to become involved. If you would like to become a member of an existing working group, as have over four hundred people, you must complete an application which is available from the ESTA office. Your application is subject to approval by the working group and you will be required to actively participate in the work of the group. This includes responding to letter ballots and attending meetings. Membership in ESTA is not a requirement, but there is an annual participation fee. A participation fee fund is available to help those who find the fee is an impediment to their participation due to their financial situation. You can also become involved by requesting that the TSC develop a standard or a recommended practice in an area of concern to you.

The Control Protocols Working Group, which authored this Standard, consists of a cross section of entertainment industry professionals representing a diversity of interests. ESTA is committed to developing consensus-based standards and recommended practices in an open setting.

Investors in Innovation

The Technical Standard Program is financially supported by companies and individuals who make undirected donations to the TSP. Contributing companies and individuals who have helped fund the TSP are recognized as “Investors in Innovation.” The Investors in Innovation when this standard was approved by ANSI on 18 May 2017 include these companies and individuals:

VISIONARY LEADERS ($50,000 & up)
ETC
ProSight Specialty Insurance

VISIONARY ($10,000 & up; >100 employees/members)
Chauvet Professional
Columbus McKinnon Entertainment Technology
Martin Professional
Robe
United States Institute for Theatre Technology
VER
Walt Disney Parks and Resorts

VISIONARY ($5,000 & up; 20–100 employees/members)
Altman Lighting, Inc.
German Light Products
High End Systems
JR Clancy
McLaren Engineering Group
Stage Rigging
TMB
Tyler Truss Systems, Inc.

VISIONARY ($500 & up; <20 employees/members)
B-Hive Industries, Inc.
Scott Blair
Boston Illumination group
Candela Controls Inc.
Clark Reder Engineering
Tracey Cosgrove & Mark McKinney
Doug Fleenor Design
EGI Event Production Services
Entertainment Project Services
Neil Huff
Hughston Engineering Inc.
Interactive Technologies
Jules Lauve
Brian Lawlor
Limelight Productions, Inc.
John T. McGraw
Mike Garl Consulting
Mike Wood Consulting
Reed Rigging
Reliable Design Services
Alan Rowe
David Saltiel
Sapsis Rigging Inc.
Stageworks
Dana Taylor
Steve Terry
Theatre Projects
Theatre Safety Programs
Tobins Lake Sales Theatrical Supply
Vertigo
Steve A. Walker & Associates
WNP Services

INVESTOR ($3,000–$9,999; >100 employees/members)
Barbizon Electric
Golden Sea Professional Equipment Limited
IATSE Local 891
Lex
NAMM
Rosco Laboratories
Texas Scenic Company

INVESTOR ($1,500–$4,999; 20–100 employees/members)
American Society of Theatre Consultants
City Theatrical Inc.
InterAmerica Stage, Inc.
Lycian Stage Lighting
Morpheus Lights
Niscon Inc.
Syracuse Scenery and Stage Lighting
XSF Xtreme Structures and Fabrication

INVESTOR ($200–$499; <20 employees/members)
About the Stage
Benjamin Cohen
Tony Giovannetti
Indianapolis Stage Sales & Rentals, Inc.
Jason Kyle
Eric Loader
LuciTag
Lumenradio AB
Moss LED
Nudelta Digital
Project SSSHH Incorporated
Stephen Vanciel

SUPPORTER (<$3,000; >100 employees/members)

Ian Foulds, IATSE Local 873
Harlequin Floors
IATSE Local 80
PSAV

SUPPORTER (<$1,500; 20–100 employees/members)

Aerial Arts
Blizzard Lighting, LLC
Creative Stage Lighting
Geiger Engineers
H&H Specialties
High Output
InCord
iWeiss
Oasis Stage Werks
Serapid
Stage Equipment & Lighting
Stagemaker
Thermotex Industries, Inc.
Tomcat
Total Structures
Ultratec Special Effects
Vincent Lighting Systems

SUPPORTER (<$200; <20 employees/members)

AC Power Distribution
Milton Davis
Peter Donovan
Pat Grenfell
Mitch Hefter
Bill Hektner
Alan Hendrickson
Hoist Sales and Services
Beverly and Tom Inglesby
Intensity Advisors
JSAV
Eddie Kramer
Michael Lay
John Musarra
Shawn Nolan
Lizz Pittsley
Phil Reilly
Robert Scales
Charles Scott
Michael Skinner
Skjonberg Controls Inc.
Studio T+L, LLC
John Szewczuk
Teclumen
Theta Consulting
Tracy Underhill
Ken Vannice ☞
Robert L. Williams

☞ Planned Giving donor

Contact Information

Technical Standards Manager
Karl G. Ruling
ESTA
630 Ninth Avenue, Suite 609
New York, NY 10036
USA
1-212-244-1505 x703
karl.ruling@esta.org

Assistant Technical Standards Manager
Erin Grabe
ESTA
630 Ninth Avenue, Suite 609
New York, NY 10036
USA
1-212-244-1505 x606
erin.grabe@esta.org

Technical Standards Council Co-chairpersons
Mike Garl
Mike Garl Consulting LLC
Phone: 1 865-389-4371
mike@mikegarlconsulting.com

Mike Wood
Mike Wood Consulting LLC
Phone: 1 512-288-4916
Fax: 1 866-674-2179
mike@mikewoodconsulting.com

Control Protocols Working Group Chairperson
Michael Lay
Philips Color Kinetics
3 Burlington Woods Drive
Burlington, MA 01803
USA
1-781-418-9145
michael.lay@philips.com

Acknowledgments

The Control Protocols Working Group members when the motion was made to reaffirm this standard at the working group's meeting on 21 January 2017 are shown below.

Voting members:

Daniel W. Antonuk; Electronic Theatre Controls, Inc.; MP
Robert Bell; Acuity Brands Inc.; MP
Marcus Bengtsson; LumenRadio AB; MP
Scott M. Blair; Full Throttle Films/VER; DR
Brent Boulnois; Candela Controls, Inc.; DE
Ian Campbell; Doug Fleenor Design, Inc.; MP
Milton Davis; Doug Fleenor Design, Inc.; MP
Adam De Witt; Adept Anomaly; U
Gary Douglas; Acuity Brands Inc.; MP
Bill Ellis; Candela Controls, Inc.; DE
Andrew Frazer; Stellascapes.com; MP
Robert Goddard; Goddard Design Co.; MP
Mitch Hefter; USITT; U
Julian Hoare; Tait Towers Manufacturing LLC; MP
Jeremy Hochman; Full Throttle Films/ VER; DR
Jon Hole; Eaton; MP
Maurits van der Hoorn; Acuity Brands Inc.; MP
John Huntington; I.A.T.S.E. Local 1; U
Michael Karlsson; LumenRadio AB; MP
Sam Kearney; Electronic Theatre Controls, Inc.; MP
Paul Kleissler; City Theatrical, Inc.; MP
Edwin S. Kramer; I.A.T.S.E. Local 1; U
Ulrich Kunkel; E3 Engineering & Education for Entertainment GmbH; U
Roger Lattin; I.A.T.S.E. Local 728; U
Michael Lay; Philips Lighting; MP
Joshua Liposky; Lex TM3; CP
Dan Lisowski; University of Wisconsin - Madison; DE
Kevin Loewen; Acuity Brands Inc.; MP
Jim Love; Tait Towers Manufacturing LLC; MP
Tyrone Mellon Jr.; Lex TM3; CP
Daniel Murfin; Royal National Theatre; U
Simon Newton; Open Lighting Project; G
Maya Nigrosh; Electronic Theatre Controls, Inc.; MP
Kimberly Corbett Oates; Schuler Shook; DE
Jim Ohrberg; Candela Controls, Inc.; DE
Claude Ostyn; Full Throttle Films/ VER; DR
Edward A. (Ted) Paget; Electronic Theatre Controls, Inc.; MP
Jason Potterf; Cisco; MP
Alan M. Rowe; I.A.T.S.E. Local 728; U
Larry Schoeneman; DesignLab Chicago, Inc.; DR
Steve Terry; Electronic Theatre Controls, Inc.; MP
Ken Vannice; Ken Vannice LLC; G
Peter Willis; Howard Eaton Lighting Ltd.; CP

Observer (non-voting) members:

Christian Allabauer; G
Matthew Ardine; IATSE Local 728; U
Robert Barbagallo; Solotech Inc.; U
Paul Beasley; Walt Disney Company; U
Lee J. Bloch; Bloch Design Group, Inc.; G

Javid D. Butler; Integrated Theatre, Inc.; CP
Justyn Butler; JBOTS; CP
Jean-Francois Canuel; A.C. Lighting Ltd.; CP
Steve Carlson; High Speed Design, Inc.; MP
Jon Chuchla; Audio Visual Systems, Inc.; G
Edward R. Condit; OSRAM Licht AG; G
Jeremy Day; Lumenpulse Lighting Inc.; MP
Larry Dew; W.A. Benjamin Electric Co.; DE
Rich Dionne; Purdue University; DE
Tucker Downs; Tucker Downs; U
Hamish Dumbreck; James Embedded Systems Engineering; MP
James Eade; ABTT; G
Paul K. Ericson; Stantec; DE
Trevor Forrest; Helvar Lighting Control; MP
Philip Gartner; AusChristmasLighting; U
Jerry Gorrell; Theatre Safety Programs; G
Sean Harding; Port Lighting Systems; G
Bill Hewlett; ImageCue LLC; MP
Jim Holladay; Luxence; G
Wayne David Howell; Artistic Licence Holdings; DE
Eric Johnson; Eric Johnson; G
Rob Johnston; Interactive Technologies, Inc.; MP
Jonathan Kemble; Barco; MP
Jason Kyle; JPK Systems Ltd.; MP
Hans Leiter; Electronic Theatre Controls, Inc.; MP
Jon Lenard; Applied Electronics; MP
Sang-Kyu Lim; Electronics and Telecommunications Research Institute; G
John Mehltretter; Lehigh Electric Products Co.; MP
John Musarra; John Musarra; U
Danilo Oliveira; Chauvet Lighting; MP
Gary Pritchard; LSC Lighting Systems PTY Ltd; MP
Charles Reese; Production Resource Group; DR
Yngve Sandboe; Sand Network Systems, Inc.; MP
Nicolai Gubi Schmidt; Gobo & Highlight A/S; DR
Ford Sellers; Chauvet Lighting; MP
Sean Sill; Sean Sill; CP
Ralph Stillinger; Philips Lighting; MP
Christopher Tilton; Westlake Reed Leskosky; DE
Robert Timmerman; Philips Lighting; MP
James Tomlinson; Team Tomlinson; G
Tracy Underhill; Triple C Lighting & Controls; G
Steve Unwin; Pulsar Ltd.; MP
Carlo Venturati; Clay Paky S.P.A.; MP
Will Wagner; Carallon Ltd.; MP
Oliver Waits; Avolites Ltd.; MP
Colin Waters; TMB; DR
Ralph Weber; ENDL Texas; G
Loren Wilton; Showman Systems; CP
David Yellin; LightMinded Industries, Inc.; MP
Jeong Sik Yoo; Korea Testing Laboratory / Theatre Safety Center; DE

Interest category codes:
CP = custom-market producer DE = designer
DR = dealer rental company G = general interest
MP = mass-market producer U = user

Table of Contents

Introduction

The ANSI E1.20 Remote Device Management Protocol (RDM) permits intelligent bi-directional communication between devices from multiple manufacturers utilizing a modified DMX512 data link. RDM is an EF(Enhanced Functionality) 1.0 implementation of ANSI E1.11 (DMX512-A).

RDM permits a console or other controlling device to discover and then configure, monitor, and manage intermediate and end-devices connected through a DMX512 network. RDM provides for intelligent control of devices on a DMX512 network, which has not been previously available outside of proprietary networks.

Overview

This document provides additional get/set parameter messages (PIDs) for use with the ANSI E1.20 Remote Device Management protocol. Many messages in this document are intended for, but not limited to, use with dimming systems.

The RDM standard can be implemented in DMX512 dimmers, to allow a controller to discover them, set the DMX512 addresses (either of the entire device or of each output separately using sub-device messaging), and to monitor sensors, such as temperatures.

This document defines additional message capabilities to access configuration parameters commonly found in many systems.

1 Normative References

ANSI E1.20-2006 *Entertainment Technology* -- Remote Device Management over USITT DMX512

ANSI E1.11-2008 *Entertainment Technology -- USITT DMX512-A -- Asynchronous Serial Digital Data Transmission Standard for Controlling Lighting Equipment and Accessories.*

ESTA
630 Ninth Avenue, Suite 609
New York, NY 10036
1-212-244-1505
http://tsp.esta.org/tsp/about/index.html

ISO/IEC 646 *Information Technology - ISO 7-bit Coded Character Set for Information Interchange*

ISO 639-1 *Codes for the representation of names of languages – Part 1: Alpha-2 code*

IEC
International Electrotechnical Commission
PO Box 131
3 rue de Varembe
1211 Geneva 20
Switzerland
+41 22 919 02 11
www.iec.ch

ISO
International Organization for Standardization
1, Rue de Varembe
Case Postale 56
CH-1211 Geneva 20
Switzerland
+41 22 74 901 11
www.iso.ch

2 General Sub-Device Handling

Refer to ANSI E1.20 Section 9 for information on Sub-Device usage. This document does not change or modify the requirements stated in ANSI E1.20. Requirements stated in this document are in addition to the stated ANSI E1.20 requirements.

Many parameter messages in this document are intended for use with products that contain Sub-Devices. An example of such a product would be a dimmer rack comprising dimmer modules, where each dimmer module is exposed as a Sub-Device.

When required, setting a parameter on all Sub-Devices at once shall be done using the SUB_DEVICE_ALL_CALL Sub-Device ID to address the message to all Sub-Devices.

Sending messages addressed to the root device as a means to globally set properties of the sub-devices shall not be allowed unless specifically stated otherwise.

Sub-devices should always declare all their supported parameters in their list of Supported Parameters. This includes those PIDs that are marked as “Required” in the required column of the RDM Parameter ID Defines in Table A-3 of ANSI E1.20.

Implementers are strongly encouraged to support the DEVICE_INFO and SUPPORTED_PARAMETERS PIDs in sub-devices. Without these PIDs, a controller has no way to determine the basic capabilities of the responder's sub-devices.

3 General Parameter Messages

3.1 General

These parameter messages are intended for general purpose use across any type of device and not limited to any specific class of products.

The GET: SUPPORTED_PARAMETERS message defined in ANSI E1.20 standard describes how responders are required to expose their list of supported parameters.

A controller can determine if a device supports the new messages defined in the sections below by using the existing GET: SUPPORTED_PARAMETERS message defined in the ANSI E1.20 standard.

3.2 Get/Set Identify Mode (IDENTIFY_MODE)

This parameter is used to get or set the RDM Identify Mode.

This parameter allows devices to have different Identify Modes for use with the IDENTIFY_DEVICE message.

If a device is currently in the IDENTIFY on state, then a change in IDENTIFY_MODE shall be automatically reflected in the device’s behavior.

Controllers wishing to provide a consistent behavior among Sub-devices may use SUB_DEVICE_ALL_CALL to change all sub-devices simultaneously.

Controller: (GET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200
(CC) GET_COMMAND	(PID) IDENTIFY_MODE	(PDL) 0x00
(PD) Not Present		

Response:

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) IDENTIFY_MODE	(PDL) 0x01
(PD) Identify Mode		

Controller: (SET)

(Port ID) 0x01 – 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200 or 0xFFFF
(CC) SET_COMMAND	(PID) IDENTIFY_MODE	(PDL) 0x01
(PD) Identify Mode		

Response (SET):

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) SET_COMMAND_RESPONSE	(PID) IDENTIFY_MODE	(PDL) 0x00
(PD) Not Present		

Data Description:

Identify Mode:
A value of 0x00 represents a “Quiet Identify” mode. This would typically be used for flashing a front panel indicator or display to visually identify the device discretely in a show situation with an audience present.

A value of 0xFF represents a “Loud Identify” mode. This would typically be used for a highly visual indication such as strobing a lighting fixture or outputting fog in a non-show situation without an audience present.

Values in the range of 0x01-0xFE are not currently defined.

3.3 Get/Set DMX512 Block Address (DMX_BLOCK_ADDRESS)

This parameter provides a mechanism for block addressing the DMX512 start address of sub-devices.

Sub-devices implementations, such as dimmer racks, are often composed of an array of sub-devices (i.e. dimmer modules) that allow a DMX512 start address to be set for the sub-device. Often it is desirable to linearly address the sub-devices to consume a contiguous block of DMX512 slots. This message provides a convenient way of accomplishing this without the need of sending a SET_COMMAND message to address each sub-device.

Since this message globally affects all sub-devices within a device, it shall only be sent to the root device of the product.

Controller: (GET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root)
(CC) GET_COMMAND	(PID) DMX_BLOCK_ADDRESS	(PDL) 0x00
(PD) Not Present		

Response:

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_ RESPONSE	(PID) DMX_BLOCK_ADDRESS	(PDL) 0x04
(PD) Total Sub-Device Footprint (16-bit) Base DMX512 Address 1-512, 0xFFFF (16-bit)		

Controller: (SET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root)
(CC) SET_COMMAND	(PID) DMX_BLOCK_ADDRESS	(PDL) 0x02
(PD) Base DMX512 Address 1-512 (16-bit)		

Response:

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) SET_COMMAND_ RESPONSE	(PID) DMX_BLOCK_ADDRESS	(PDL) 0x00
(PD) Not Present		

The Total Sub-Device Footprint shall return the total combined DMX512 footprint (number of consecutive DMX512 slots required) of all the sub-devices within the device. The footprint of the root device shall not be included within this footprint field.

The GET_COMMAND returns the current base DMX512 start address for the array of sub-devices. This is equivalent to the DMX512 Start Address of the first sub-device if the sub-devices are all linearly addressed as a contiguous block. If the sub-devices are not currently linearly addressed as a contiguous block then this field shall be set to 0xFFFF in the response message.

The SET_COMMAND shall set the DMX512 address for the first sub-device to the specified address and the device shall automatically address each sub-device incrementally accounting for the footprint size of each sub-device.

This message shall not have any effect on the DMX512 Start Address for the root device, only the sub-devices.

3.4 Get/Set DMX512 Fail Mode (DMX_FAIL_MODE)

This parameter defines the behavior of the device when the DMX512 control signal is lost.

A scene that is triggered by a DMX512 Loss of Signal condition should ignore the Wait Time stored using the CAPTURE_PRESET PID from ANSI E1.20 and instead use the Hold Time included with this PID.

The setting is usually per device, but may be supported at the sub-device level.

Setting DMX512 Fail Mode for all sub-devices can be done by using the SUB_DEVICE_ALL_CALL to address the message to all sub-devices.

If an attempt is made to set DMX512 Fail Mode for a sub-device and the device only supports this function globally (i.e. root-device), it shall respond with a NACK with a NACK Reason Code of NR_UNKNOWN_PID.

Controller: (GET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001- 0x0200
(CC) GET_COMMAND	(PID) DMX_FAIL_MODE	(PDL) 0x00
(PD) Not Present		

Response: (GET)

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) DMX_FAIL_MODE	(PDL) 0x07
(PD) Scene # (16-bit) Loss of Signal Delay (16-bit) Hold Time (16-bit) Level		

Controller: (SET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001- 0x0200 or 0xFFFF
(CC) SET_COMMAND	(PID) DMX_FAIL_MODE	(PDL) 0x07
(PD) Scene # (16-bit) Loss of Signal Delay Time (16-bit) Hold Time (16-bit) Level		

Response: (SET)

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) SET_COMMAND_RESPONSE	(PID) DMX_FAIL_MODE	(PDL) 0x00
(PD) Not Present		

Data Description:

Scene #:
The scene number (see ANSI E1.20 Table A-7) for the device to transition to when the DMX512 signal is lost. The fade time stored with the preset is used when performing this function. A scene number of PRESET_PLAYBACK_OFF shall cause the device to fade or switch to the value indicated in the Level field when the DMX512 signal has been lost.

If the device does not support the requested scene number, it shall respond with a NACK with reason NR_DATA_OUT_OF_RANGE.

Loss of Signal Delay Time:
This field is the amount of time that the device shall delay playing the scene specified from the point when the DMX512 signal is lost. Times are specified in tenths of a second. The DMX512 signal is considered to be lost when no null start code packets have been received for a period of greater than 1.25 seconds (See ANSI E1.11 Section 9.2). A value of 0xFFFF shall be used to specify an infinite loss of signal delay

time. The result of an infinite loss of signal delay time means that the device shall hold its last look until the DMX512 signal is restored.

If the device receives a Loss of Signal Delay time that is outside the minimum or maximum time allowed by the device, then it shall set the field to the minimum or maximum accordingly.

The minimum, maximum, and support for infinite delay times may be exposed using the PRESET_INFO parameter message.

Hold Time:
This field sets the amount of time that the device shall playback the specified scene before going to black, entering shutdown, or any other device-specific loss of DMX512 condition. Times are specified in tenths of a second. A value of 0xFFFF shall be used to specify an infinite hold time. The result of an infinite hold time is that the device shall continue playing the scene indefinitely.

If the device receives a Hold Time that is outside the minimum or maximum time allowed by the device, then it shall set the field to the minimum or maximum accordingly.

The minimum, maximum, and support for infinite hold times may be exposed using the PRESET_INFO parameter message.

Level:
When a scene number is selected to be played, this field sets the proportional intensity for the scene. If it is at full (0xFF), then the scene shall be played as recorded. Otherwise, it scales the level of the scene proportionally.

When a scene number of PRESET_PLAYBACK_OFF is selected then the level field shall set an overall level for devices that support master level functionality.

An example is for a situation where a loss of DMX condition needs to turn all dimmers to a set level, but the dimmers may not have scene playback capabilities. Using the scene number PRESET_PLAYBACK_OFF, setting the level field of 50% would bring all dimmers to the set level of 50%.

Some applications may require loss of DMX condition processing as part of an overall emergency lighting or safety consideration strategy. Use of the Level field to force dimmer outputs may not be appropriate in all situations and is not required in such circumstances. Such considerations are beyond the scope of this document.

3.5 Get/Set DMX512 Startup Mode (DMX_STARTUP_MODE)

This parameter defines the behavior of the device when it starts up and the DMX512 control signal is absent. The DMX512 signal is considered absent when null start code packets are not received for a period of greater than 1.25 seconds (See ANSI E1.11 Section 9.2).

A scene that is triggered by a DMX512 control signal not being present at startup should ignore the Wait Time stored using the CAPTURE_PRESET PID from ANSI E1.20 and instead use the Hold Time included with this PID.

The setting is usually per device, but may be supported at the sub-device level.

Setting Startup Mode for all sub-devices can be done by using the SUB_DEVICE_ALL_CALL to address the message to all sub-devices.

If an attempt is made to set Startup Mode for a sub-device and the device only supports this function globally (i.e. root-device), it shall respond with a NACK with a NACK Reason Code of NR_UNKNOWN_PID.

Controller: (GET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001- 0x0200
(CC) GET_COMMAND	(PID) DMX_STARTUP_MODE	(PDL) 0x00
(PD) Not Present		

Response: (GET)

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) DMX_STARTUP_MODE	(PDL) 0x07
(PD) Scene # (16-bit) Startup Delay (16-bit) Hold Time (16-bit) Level		

Controller: (SET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001- 0x0200 or 0xFFFF
(CC) SET_COMMAND	(PID) DMX_STARTUP_MODE	(PDL) 0x07
(PD) Scene # (16-bit) Startup Delay Time (16-bit) Hold Time (16-bit) Level		

Response: (SET)

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) SET_COMMAND_RESPONSE	(PID) DMX_STARTUP_MODE	(PDL) 0x00
(PD) Not Present		

Data Description:

Scene #:
The scene number (see ANSI E1.20 Table A-7) for the device to fade to at startup if the DMX512 control signal is absent. The fade time stored with the preset is used when performing this function. A scene number of PRESET_PLAYBACK_OFF shall cause the device to fade or switch to the value indicated in the Level field when the DMX512 signal is absent at startup.

If the device does not support the requested scene number, it shall respond with a NACK with reason NR_DATA_OUT_OF_RANGE.

Startup Delay Time:
This field sets the amount of time that the device shall delay playing the scene specified from startup. Times are specified in tenths of a second.

If the device receives a Startup Delay time that is outside the minimum or maximum time allowed by the device, then it shall set the field to the minimum or maximum accordingly.

The minimum and maximum times may be exposed using the PRESET_INFO parameter message.

Hold Time:
This field sets the amount of time that the device shall playback the specified scene before going to black, entering shutdown, or any other device-specific behavior. Times are specified in tenths of a second. A value of 0xFFFF shall be used to specify an infinite hold time. The result of an infinite hold time is that the device shall continue playing the scene until such time as a DMX512 control signal is detected.

If the device receives a Hold Time that is outside the minimum or maximum time allowed by the device, then it shall set the field to the minimum or maximum accordingly.

The minimum, maximum, and support for infinite hold times may be exposed using the PRESET_INFO parameter message.

Level:
This field sets the proportional intensity for the scene. If it is at full (0xFF), then the scene shall be played as recorded. Otherwise, it scales the level of the scene proportionally.

When a scene number of PRESET_PLAYBACK_OFF is selected than the level field shall set an overall level for the device for devices that support master level functionality.

An example is for a situation where a loss of DMX condition needs to turn all dimmers to a set level, but the dimmers may not have scene playback capabilities. Using the scene number PRESET_PLAYBACK_OFF, setting the level field of 50% would bring all dimmers to the set level of 50%.

Some applications may require loss of DMX condition processing as part of an overall emergency lighting or safety consideration strategy. Use of the Level field to force dimmer outputs may not be appropriate in all situations and is not required in such circumstances. Such considerations are beyond the scope of this document.

3.6 Get/Set Power-On Self Test (POWER_ON_SELF_TEST)

This parameter is used to get or set the Power-On Self Test mode parameter.

This allows devices to enable or disable a power-on self test mode that executes automatically on power up.

When enabled, the power-on self test shall execute on power-on before any defined start-up behaviors or normal operation begins.

Controller: (GET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200
(CC) GET_COMMAND	(PID) POWER_ON_SELF_TEST	(PDL) 0x00
(PD) Not Present		

Response:

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) POWER_ON_SELF_TEST	(PDL) 0x01
(PD) Disabled/Enabled (0/1)		

Controller: (SET)

(Port ID) 0x01 – 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200 or 0xFFFF
(CC) SET_COMMAND	(PID) POWER_ON_SELF_TEST	(PDL) 0x01
(PD) Disabled/Enabled (0/1)		

Response (SET):

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) SET_COMMAND_RESPONSE	(PID) POWER_ON_SELF_TEST	(PDL) 0x00
(PD) Not Present		

3.7 Get/Set Lock State (LOCK_STATE)

This parameter is used to determine the lock state for devices that support locking. A lock, when applied, can provide a variable level of access/change protection.

The locking mechanism is designed to deter tampering and is not intended to provide absolute security.

For any given lock state value, the GET: LOCK_STATE_DESCRIPTION message shall be used to get a text description of what functionality that lock state provides.

Responders that support LOCK_STATE shall also support the LOCK_STATE_DESCRIPTION message.

When the SET_COMMAND functionality of any PID is locked for a device, with LOCK_STATE or otherwise, it shall respond to SET_COMMAND messages with a NACK with a NACK Reason Code of NR_WRITE_PROTECT.

Controller: (GET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200
(CC) GET_COMMAND	(PID) LOCK_STATE	(PDL) 0x00
(PD) Not Present		

Response:

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) LOCK_STATE	(PDL) 0x02
(PD) Current Lock State | # of Lock States		

Controller: (SET)

(Port ID) 0x01 – 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200 or 0xFFFF
(CC) SET_COMMAND	(PID) LOCK_STATE	(PDL) 0x03
(PD) PIN Code (16-bit) | Lock State		

Response (SET):

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) SET_COMMAND_RESPONSE	(PID) LOCK_STATE	(PDL) 0x00
(PD) Not Present		

Data Description:

Current Lock State:
This is the current lock state that is active in the device. A returned value of 0x00 shall mean that the device is unlocked.

# of Lock States:
The total number of locked states that the device is capable of providing, in addition to the unlocked state 0x00. Lock states shall be consecutively numbered within the responder starting from 0x01 (0x00 is unlocked).

Lock State:
A value of 0x00 is used to unlock a device. Values between 0x01 and the declared # of Lock States may be used to lock a device.

PIN Code:
The current PIN code is provided for devices that require validating the PIN code before accepting a change to the lock state. For devices that do not require a valid PIN code before changing the lock state, this field shall be ignored.

If the device does require a PIN code and determines that the Current PIN code is incorrect, then the device shall send a NACK with a NACK Reason Code of NR_DATA_OUT_OF_RANGE.

See Section 3.9 for PIN Code range restrictions.

3.8 Get Lock State Description (LOCK_STATE_DESCRIPTION)

This parameter is used to get a descriptive ASCII text label for a given lock state. The label may be up to 32 characters.

Controller: (GET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200
(CC) GET_COMMAND	(PID) LOCK_STATE_DESCRIPTION	(PDL) 0x01
(PD) Lock State Requested		

Response:

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) LOCK_STATE_DESCRIPTION	(PDL) 1 – 33 (1 + Number of characters sent)
(PD) Lock State Requested ASCII Text field of variable size		

The Response Data contains the Lock State Requested along with up to 32 characters of description.

Valid values for “Lock State Requested” are between 0x01 and the # of Lock States reported by the GET: LOCK_STATE PID.

3.9 Get/Set Lock PIN (LOCK_PIN)

This parameter is used to get and set the PIN code for devices that support locking. The lock state is set using the LOCK_STATE message.

The PIN format and length is deliberately kept simple to ensure interoperability with devices that may have limited UI and display capabilities.

Manufacturers are reminded that a lost PIN code may render a device inaccessible to the user. Lost PIN codes and ways to bypass locking on locked devices are beyond the scope of this standard.

Manufacturers who require a higher level of security, or more complex PINs may use manufacturer-specific parameter messages to implement those features.

Controller: (GET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200
(CC) GET_COMMAND	(PID) LOCK_PIN	(PDL) 0x00
(PD) Not Present		

Response:

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) LOCK_PIN	(PDL) 0x02
(PD) Current PIN code (16-bit)		

Controller: (SET)

(Port ID) 0x01 – 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200 or 0xFFFF
(CC) SET_COMMAND	(PID) LOCK_PIN	(PDL) 0x04
(PD) New PIN code (16-bit) Current PIN code (16-bit)		

Response (SET):

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) SET_COMMAND_RESPONSE	(PID) LOCK_PIN	(PDL) 0x00
(PD) Not Present		

Some devices may disallow getting the current PIN code using this message. For devices that protect against the RDM Controller retrieving the current PIN code, any GET_COMMAND messages shall send a NACK with a NACK Reason of NR_UNSUPPORTED_COMMAND_CLASS.

PIN Codes shall be in the range of 0x0000-0x270F (i.e. 0000-9999). The recommended default PIN code for a device is 0x0000. If a device receives a PIN code outside of this range, then it shall send a NACK with a NACK Reason Code of NR_FORMAT_ERROR.

In the SET_COMMAND message the Current PIN code is provided for devices that choose to require a valid PIN code before accepting a change. Devices that do not require this may ignore the value for the Current PIN code in the SET_COMMAND.

If the device determines that the Current PIN code is incorrect, then the device shall send a NACK with a NACK Reason Code of NR_DATA_OUT_OF_RANGE.

3.10 Get/Set Burn-In (BURN_IN)

This parameter provides a mechanism for devices that require specified burn-in times.

In order for fluorescent lamps to operate properly with all types of fluorescent dimming ballasts they must be operated continuously at full output for a manufacturer recommended period of time (BURN_IN).

This parameter will allow users to set a burn-in time for dimmers controlling fluorescent ballasts after changing lamps, for example.

The Get message shall return the remaining time on the burn in. The Set message shall set the burn in time and start the burn in process. Setting a burn-in time of 0x00, shall abort any burn-in process running.

Controller: (GET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200
(CC) GET_COMMAND	(PID) BURN_IN	(PDL) 0x00
(PD) Not Present		

Response:

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) BURN_IN	(PDL) 0x01
(PD) Hours Remaining (0-255)		

Controller: (SET)

(Port ID) 0x01 – 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200 or 0xFFFF
(CC) SET_COMMAND	(PID) BURN_IN	(PDL) 0x01
(PD) Hours (0-255)		

Response (SET):

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) SET_COMMAND_RESPONSE	(PID) BURN_IN	(PDL) 0x00
(PD) Not Present		

4 Dimmer Parameter Messages

4.1 General

RDM Dimmer Parameter Messages are used to set configuration of and get status information from the dimming system.

The GET: SUPPORTED_PARAMETERS message defined in ANSI E1.20 standard describes how responders are required to expose their list of supported parameters.

A controller can determine if a device supports the new messages defined in the sections below by using the existing GET: SUPPORTED_PARAMETERS message defined in the ANSI E1.20 standard.

4.2 Intensity Levels

All intensity level fields are transmitted as 16-bit intensity values. Devices that make use of fewer than 16 bits of intensity values shall use the most significant bits of the level field and truncate or round any lower order bit for SET_COMMAND messages and pad lower order bits as 0 for GET_COMMAND messages.

4.3 Get Dimmer Info (DIMMER_INFO)

This parameter is used to retrieve a variety of dimmer related information that describes the capabilities of the dimmer.

Controller: (GET) Dimmer Info

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001- 0x0200
(CC) GET_COMMAND	(PID) DIMMER_INFO	(PDL) 0x00
(PD) Not Present		

Response: (GET)

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) DIMMER_INFO	(PDL) 0x0B
(PD) Minimum Level – Lower Limit (16-bit) Minimum Level – Upper Limit(16-bit) Maximum Level – Lower Limit (16-bit) Maximum Level – Upper Limit (16-bit) Number of Supported Curves Levels Resolution Minimum Level - Split levels Supported		

Data Description:

Minimum Level – Lower Limit
This field indicates the lowest value that MINIMUM_LEVEL can be set to. Devices not implementing MINIMUM_LEVEL shall set this field to 0x0000.

Minimum Level – Upper Limit
This field indicates the highest value that MINIMUM_LEVEL can be set to. Devices not implementing MINIMUM_LEVEL shall set this field to 0x0000.

Maximum Level – Lower Limit
This field indicates the lowest value that MAXIMUM_LEVEL can be set to. Devices not implementing MAXIMUM_LEVEL shall set this field to 0xFFFF.

Maximum Level – Upper Limit
This field indicates the highest value that MAXIMUM_LEVEL can be set to. Devices not implementing MAXIMUM_LEVEL shall set this field to 0xFFFF.

Number of Supported Curves:
This field indicates the total number of curves the device supports.

Levels Resolution:
This field indicates the number of bits that the device uses in level values in RDM messages that use 16-bit level fields. The allowable range for this field is 0x01-0x10 (1-16).

Minimum Level – Split levels Supported
This field indicates whether the device supports split levels (Increasing and Decreasing) for MINIMUM_LEVEL. 0x00 means the device does not support split levels, 0x01 means the device does support split levels. Devices not implementing MINIMUM_LEVEL shall set this field to 0x00.

4.4 Get/Set Minimum Level (MINIMUM_LEVEL)

Minimum Level sets the lowest level that the output may go to in response to the control signal - DMX512, Preset Playback or otherwise. By setting the **On Below Minimum** field, this can be used to provide Preheat functionality for incandescent lamps.

Two Minimum Levels are defined – Increasing and Decreasing. This allows devices to switch on their output at one level, and switch off their output at a different level. Split levels are useful for controlling dimmable fluorescent lamps. Split level functionality is optional as indicated by the **Minimum Level – Split levels Supported** field in the DIMMER_INFO message. Devices not supporting split level functionality shall use the **Minimum Level – Increasing** field to set the minimum level.

The upper and lower limits for Minimum Level are indicated in the DIMMER_INFO message.

Controller: (GET) Minimum Level

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001- 0x0200
(CC) GET_COMMAND	(PID) MINIMUM_LEVEL	(PDL) 0x00
(PD) Not Present		

Response: (GET)

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) MINIMUM_LEVEL	(PDL) 0x05
(PD) Minimum Level – Increasing (16-bit) Minimum Level – Decreasing (16-bit) On Below Minimum		

Controller: (SET) Minimum Level

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001- 0x0200 or 0xFFFF
(CC) SET_COMMAND	(PID) MINIMUM_LEVEL	(PDL) 0x05
(PD)		

(PD)
Minimum Level – Increasing (16-bit)
Minimum Level – Decreasing (16-bit)
On Below Minimum

Response: (SET)

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) SET_COMMAND_RESPONSE	(PID) MINIMUM_LEVEL	(PDL) 0x00
(PD) Not Present		

Data Description:

Minimum Level – Increasing:
This field sets the minimum level that an output may go to when the control signal for the output is increasing (fading up).

Minimum Level – Decreasing:
This field sets the minimum level that an output may go to when the control signal for the output is decreasing (fading down). Devices that do not support split levels shall ignore this field in a SET message, and report the Minimum Level in response to a GET message.

On Below Minimum:
If the control signal for the output is below the minimum level, then TRUE (0x01) means the output holds the minimum level and FALSE (0x00) means the output switches off. Devices that do not support configuration of this option shall ignore this field in a SET message, and report their fixed behavior in response to a GET.

Devices that only support a fixed minimum level (e.g. a simple Preheat off/on setting) shall use any non-zero value in a SET to enable that minimum level. In this case, the response to a GET shall indicate the fixed minimum level.

4.5 Get/Set Maximum Level (MAXIMUM_LEVEL)

Maximum Level sets the highest level that the output may go to in response to the control signal - DMX512, Preset Playback or otherwise. This can be used to provide Topset functionality.

The upper and lower limits for Maximum Level are indicated in the DIMMER_INFO message.

Controller: (GET) Maximum Level

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200
(CC) GET_COMMAND	(PID) MAXIMUM_LEVEL	(PDL) 0x00
(PD) Not Present		

Response:

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) MAXIMUM_LEVEL	(PDL) 0x02
(PD) Maximum Level (16-bit)		

Controller: (SET) Maximum Level

(Port ID) 0x01 – 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200 or 0xFFFF
(CC) SET_COMMAND	(PID) MAXIMUM_LEVEL	(PDL) 0x02
(PD) Maximum Level (16-bit)		

Response (SET):

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) SET_COMMAND_RESPONSE	(PID) MAXIMUM_LEVEL	(PDL) 0x00
(PD) Not Present		

Data Description:

Maximum Level:
This field sets the maximum level that an output may go to in response to the control signal. When the control signal exceeds this level, the output shall hold at the maximum level.

4.6 Get/Set Curve (CURVE)

Sometimes called dimmer laws, curves set a relationship between the control level and the output level. This is useful when matching different loads, or when matching different dimmer types. On more advanced dimmers, it may be possible to program user-defined curves. Transferring user defined curve data is beyond the scope of this standard.

This parameter is used to Get/Set the curve setting for the device.

The GET_COMMAND_RESPONSE message includes the current curve setting as well as the total number of curves available. These curves shall be consecutively numbered within the responder starting from 1.

Text descriptions can be retrieved using the CURVE_DESCRIPTION Parameter. Responders that support CURVE shall also support the CURVE_DESCRIPTION message.

Controller: (GET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200
(CC) GET_COMMAND	(PID) CURVE	(PDL) 0x00
(PD) Not Present		

Response:

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) CURVE	(PDL) 0x02
(PD) Current Curve | # of Curves		

Controller: (SET)

(Port ID) 0x01 – 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200 or 0xFFFF
(CC) SET_COMMAND	(PID) CURVE	(PDL) 0x01
(PD) Curve		

Response (SET):

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) SET_COMMAND_RESPONSE	(PID) CURVE	(PDL) 0x00
(PD) Not Present		

Some sub-devices may not support all the curves of another sub-device within the same product. For system management simplification, all sub-devices shall report an identical set of curve options, which is the combined list of all possible curves that may occur in the system. Curve numbers shall be the same across all sub-devices within a product. If a particular sub-device does not support a given curve, then it shall send respond to a SET_COMMAND message with a NACK with a NACK Reason Code of NR_DATA_OUT_OF_RANGE.

4.7 Get Curve Description (CURVE_DESCRIPTION)

This parameter is used to get a descriptive ASCII text label for a given Curve number. The label may be up to 32 characters.

Controller: (GET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200
(CC) GET_COMMAND	(PID) CURVE_DESCRIPTION	(PDL) 0x01
(PD) Curve Requested		

Response:

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) CURVE_DESCRIPTION	(PDL) 1 – 33 (1 + Number of characters sent)
(PD) Curve Requested ASCII Text field of variable size		

The Response Data contains the Curve Requested along with up to 32 characters of description.

4.8 Get/Set Output Response Time (OUTPUT_RESPONSE_TIME)

Dimmers often have a variable response time that smoothes fades that might otherwise exhibit a stepping behavior between levels. The consequence of smoothing fades using this method is that the dimmer may not turn on or off as quickly as it would without the slowed response. Dimmers with variable response times allow the user to achieve a balance between speed and smoothness in fades.

This parameter is used to get or set the response time for the device.

Controller: (GET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200
(CC) GET_COMMAND	(PID) OUTPUT_RESPONSE_TIME	(PDL) 0x00
(PD) Not Present		

Response:

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) OUTPUT_RESPONSE_TIME	(PDL) 0x02
(PD) Current Response Time | # of Response options		

Controller: (SET)

(Port ID) 0x01 – 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200 or 0xFFFF
(CC) SET_COMMAND	(PID) OUTPUT_RESPONSE_TIME	(PDL) 0x01
(PD) Response Time		

Response (SET):

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) SET_COMMAND_RESPONSE	(PID) OUTPUT_RESPONSE_TIME	(PDL) 0x00
(PD) Not Present		

The GET_COMMAND_RESPONSE message includes the response time setting as well as the total number of response settings available. These output response times shall be consecutively numbered within the range starting from 1. When multiple response times are available, the fastest time shall be represented by 1. Higher numbered Response Time values shall represent progressively longer times.

Text descriptions can be retrieved using the OUTPUT_RESPONSE_TIME_DESCRIPTION parameter. Responders that support OUTPUT_RESPONSE_TIME shall also support the OUTPUT_RESPONSE_TIME_DESCRIPTION message.

4.9 Get Response Time Description (OUTPUT_RESPONSE_TIME_DESCRIPTION)

This parameter is used to get a descriptive ASCII text label for a response time setting. The label may be up to 32 characters.

Controller: (GET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200
(CC) GET_COMMAND	(PID) OUTPUT_RESPONSE_TIME_DESCRIPTION	(PDL) 0x01
(PD) Response Time Requested		

Response:

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) OUTPUT_RESPONSE_TIME_DESCRIPTION	(PDL) 1 – 33 (1 + Number of characters sent)
(PD) Response Time Requested ASCII Text field of variable size		

The Response Data contains the Response Time Requested along with up to 32 characters of description.

4.10 Get/Set Modulation Frequency (MODULATION_FREQUENCY)

This parameter is used to get and set the modulation frequency for devices that support adjustment of the modulation frequency of their output.

The GET_COMMAND_RESPONSE message includes the current modulation frequency setting as well as the total number of settings available. These settings shall be consecutively numbered within the responder starting from 1.

Text descriptions can be retrieved using the MODULATION_FREQUENCY_DESCRIPTION Parameter. Responders that support MODULATION_FREQUENCY shall also support the MODULATION_FREQUENCY_DESCRIPTION message.

Controller: (GET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200
(CC) GET_COMMAND	(PID) MODULATION_FREQUENCY	(PDL) 0x00
(PD) Not Present		

Response:

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) MODULATION_FREQUENCY	(PDL) 0x02
(PD) Current setting | # of settings available		

Controller: (SET)

(Port ID) 0x01 – 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200 or 0xFFFF
(CC) SET_COMMAND	(PID) MODULATION_FREQUENCY	(PDL) 0x01
(PD) Modulation Frequency Setting		

Response (SET):

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) SET_COMMAND_RESPONSE	(PID) MODULATION_FREQUENCY	(PDL) 0x00
(PD) Not Present		

Some sub-devices may not support all the modulation frequencies of another sub-device within the same product. For system management simplification, all sub-devices shall report an identical set of modulation frequency options, which is the combined list of all possible modulation frequencies that may occur in the system. Setting numbers shall be the same across all sub-devices within a product. If a particular sub-device does not support a given setting, then it shall send respond to a SET_COMMAND message with a NACK with a NACK Reason Code of NR_DATA_OUT_OF_RANGE.

4.11 Get Modulation Frequency Description (MODULATION_FREQUENCY_DESCRIPTION)

This parameter is used to get a descriptive ASCII text label for a given modulation frequency setting. The label may be up to 32 characters.

Controller: (GET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001-0x0200
(CC) GET_COMMAND	(PID) MODULATION_FREQUENCY_DESCRIPTION	(PDL) 0x01
(PD) Modulation Setting Requested		

Response:

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) MODULATION_FREQUENCY_DESCRIPTION	(PDL) 5 – 37 (5 + Number of characters sent)
(PD) Modulation Setting Requested Frequency (32-bit) ASCII Text field of variable size		

Data Description:

Modulation Setting Requested:
This field indicates the modulation setting number requested from the GET_COMMAND message.

Frequency:
This field is a 32-bit unsigned integer representation of the frequency in Hz for the requested setting for machine readable applications. A frequency of 0xFFFFFFFF indicates this field is not declared.

Text Description:
The ASCII text field contains up to 32 characters of description information for the modulation setting.

5 Preset Messages

5.1 General

These parameter messages are general purpose Preset messages used to extend the functionality and capabilities of the existing Preset messages in ANSI E1.20.

The GET: SUPPORTED_PARAMETERS message defined in ANSI E1.20 standard describes how responders are required to expose their list of supported parameters.

A controller can determine if a device supports the new messages defined in the sections below by using the existing GET: SUPPORTED_PARAMETERS message defined in the ANSI E1.20 standard.

5.2 Get Preset Info (PRESET_INFO)

This parameter is used to retrieve a variety of preset related information that describes the preset capabilities of the device.

Controller: (GET) Preset Info

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001- 0x0200
(CC) GET_COMMAND	(PID) PRESET_INFO	(PDL) 0x00
(PD) Not Present		

Response: (GET)

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) PRESET_INFO	(PDL) 0x20
(PD)		

Level Field Supported
Preset Sequence Supported
Split Times Supported
DMX512 Fail Infinite Delay Time Supported
DMX512 Fail Infinite Hold Time Supported
Startup Infinite Hold Time Supported
Maximum Scene Number (16-bit)
Minimum Preset Fade Time Supported (16-bit)
Maximum Preset Fade Time Supported (16-bit)
Minimum Preset Wait Time Supported (16-bit)
Maximum Preset Wait Time Supported (16-bit)
Minimum DMX512 Fail Delay Time Supported (16-bit)
Maximum DMX512 Fail Delay Time Supported (16-bit)
Minimum DMX512 Fail Hold Time Supported (16-bit)
Maximum DMX512 Fail Hold Time Supported (16-bit)
Minimum Startup Delay Time Supported (16-bit)
Maximum Startup Delay Time Supported (16-bit)
Minimum Startup Hold Time Supported (16-bit)
Maximum Startup Hold Time Supported (16-bit)

Data Description:

Level Field Supported:
This field indicates if the Level field in the PRESET_PLAYBACK message is supported. If the device supports the use of the level field it shall respond with 0x01, otherwise it shall respond with 0x00.

Preset Sequence Supported:
This field indicates if the device is capable of playing back presets in a sequence as defined in the PRESET_PLAYBACK message. If the device supports the ability to play sequences it shall respond with 0x01, otherwise it shall respond with 0x00.

Split Times Supported:
This field indicates if the device supports split up/down times for playback of presets. If the device supports split times it shall respond with 0x01, otherwise it shall respond with 0x00. Devices that do not support split times shall use the Up Time for both fields.

Maximum Scene Number:
This field indicates the maximum preset scene number addressable by the device.

DMX512 Fail Infinite Delay Time Supported:
This field indicates if the device supports an infinite delay time for the loss of signal delay times for the DMX_FAIL_MODE message. A value of 0x01 indicates the device does support the use of infinite delay times for this message. A value of 0x00 indicates the device does not support the use of infinite delay times for this message. Devices not supporting DMX_FAIL_MODE shall report 0x00 for this field.

DMX512 Fail Infinite Hold Time Supported:
This field indicates if the device supports an infinite hold time for the DMX_FAIL_MODE message. A value of 0x01 indicates the device does support the use of infinite hold times for this message. A value of 0x00 indicates the device does not support the use of infinite hold times for this message. Devices not supporting DMX_FAIL_MODE shall report 0x00 for this field.

Startup Infinite Hold Time Supported:
This field indicates if the device supports an infinite hold time for the DMX_STARTUP_MODE message. A value of 0x01 indicates the device does support the use of infinite hold times for this message. A value of 0x00 indicates the device does not support the use of infinite hold times for this message. Devices not supporting DMX_STARTUP_MODE shall report 0x00 for this field.

Preset Minimum/Maximum Fade Times Supported:
These fields indicate the minimum and maximum fade times for a preset scene that the device can support. Times shall be reported in tenths of a second.

Preset Minimum/Maximum Wait Times Supported:
These fields indicate the minimum and maximum wait times for a preset scene that the device can support. Times shall be reported in tenths of a second.

DMX512 Fail Minimum/Maximum Delay Times Supported:
These fields indicate the minimum and maximum loss of signal delay times for the DMX_FAIL_MODE message that the device can support. Times shall be reported in tenths of a second. Devices not supporting DMX_FAIL_MODE shall report 0xFFFF for these fields.

DMX512 Fail Minimum/Maximum Hold Times Supported:
These fields indicate the minimum and maximum hold times for the DMX_FAIL_MODE message that the device can support. Times shall be reported in tenths of a second. Devices not supporting DMX_FAIL_MODE shall report 0xFFFF for these fields.

Startup Minimum/Maximum Delay Times Supported:
These fields indicate the minimum and maximum startup delay times for the STARTUP_MODE message that the device can support. Times shall be reported in tenths of a second. Devices not supporting STARTUP_MODE shall report 0xFFFF for these fields.

Startup Minimum/Maximum Hold Times Supported:
These fields indicate the minimum and maximum startup hold times for the STARTUP_MODE message that the device can support. Times shall be reported in tenths of a second. Devices not supporting STARTUP_MODE shall report 0xFFFF for these fields.

5.3 Get/Set Preset Status (PRESET_STATUS)

This parameter is used to determine if a preset scene is programmed and to retrieve the timing information stored with that scene (Get). It also allows a preset scene to be cleared or to change the timing information stored with that scene (Set).

Fade and Wait times for building sequences may also be included. Times are in tenths of a second. When timing information is not required the fields shall be set to 0x0000.

The Up Fade Time is the fade in time for the current scene and the Down Fade Time is the down fade for the previous scene or active look. The Wait Time is the time the device spends holding the current scene before proceeding to play the next scene when the presets are being played back as a sequence.

Controller: (GET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001- 0x0200
(CC) GET_COMMAND	(PID) PRESET_STATUS	(PDL) 0x02
(PD) Scene # (16-bit)		

Response: (GET)

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) PRESET_STATUS	(PDL) 0x09
(PD) Scene # (16-bit) Up Fade Time (16-bit) Down Fade Time (16-bit) Wait Time (16-bit) Programmed		

Controller: (SET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001- 0x0200 or 0xFFFF
(CC) SET_COMMAND	(PID) PRESET_STATUS	(PDL) 0x09
(PD) Scene # (16-bit) Up Fade Time (16-bit) Down Fade Time (16-bit) Wait Time (16-bit) Clear Preset		

Response: (SET)

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) SET_COMMAND_RESPONSE	(PID) PRESET_STATUS	(PDL) 0x00
(PD) Not Present		

Data Description:

Scene #:
The scene number (stored using the CAPTURE_PRESET messages – see ANSI E1.20 Section 10.11.6) range 0x0001 - 0xFFFE. Devices are not required to support all possible scene numbers, and shall respond with a NACK with reason NR_DATA_OUT_OF_RANGE if the scene number exceeds the available storage.

Programmed:
Preset Programmed states are enumerated in Table A-2. If a preset has a value of PRESET_NOT_PROGRAMMED, all timing information relating to that preset shall be set to zero in the response.

Clear Preset:
If this is set to a value of 0x00, then the preset is not cleared, the stored levels are not altered, and only the timing information is updated for the stored preset. If this is set to a value of 0x01, then the preset shall be cleared and all timing information for the stored preset shall be set to zero (the timing information in the SET shall be ignored when clearing the preset). Factory programmed presets that may not be erased shall return a NACK with a NACK Reason code of NR_WRITE_PROTECT.

5.4 Get/Set Preset Merge Mode (PRESET_MERGEMODE)

The RDM standard (ANSI E1.20 Section 10.11.7) assumes that when a preset is played with the PRESET_PLAYBACK message, that it takes precedence over the DMX512 input signal. On some devices this may not be the desired effect, and other merge modes may be offered:

This parameter is used to retrieve or change the preset merge mode.

Support for this parameter message overrides the default preset playback behavior as defined in ANSI E1.20. If a device does not declare support for this message in the SUPPORTED_PARAMETERS message, then the behavior for preset playback as defined in ANSI E1.20 shall remain in effect.

Controller: (GET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001- 0x0200
(CC) GET_COMMAND	(PID) PRESET_MERGEMODE	(PDL) 0x00
(PD) Not Present		

Response: (GET)

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) GET_COMMAND_RESPONSE	(PID) PRESET_MERGEMODE	(PDL) 0x01
(PD) Merge Mode		

Controller: (SET)

(Port ID) 0x01 - 0xFF	(Message Count) 0x00	(Sub-Device) 0x0000 (Root) or 0x0001- 0x0200 or 0xFFFF
(CC) SET_COMMAND	(PID) PRESET_MERGEMODE	(PDL) 0x01
(PD) Merge Mode		

Response: (SET)

(Response Type) ACK	(Message Count) 0x00-0xFF	(Sub-Device) Copy of Controller SD
(CC) SET_COMMAND_RESPONSE	(PID) PRESET_MERGEMODE	(PDL) 0x00
(PD) Not Present		

Data Description:

Merge Mode:
The Preset Merge Modes are enumerated in Table A-3.

Appendix A: Defined Parameters (Normative)

Table A-1: RDM Parameter ID Defines

GET Allowed	SET Allowed	RDM Parameter ID's (Slot 21-22)	Value	Comment	Required
		Category – DMX512 Setup			
✓	✓	DMX_BLOCK_ADDRESS	0x0140		
✓	✓	DMX_FAIL_MODE	0x0141		
✓	✓	DMX_STARTUP_MODE	0x0142		
		Category – Dimmer Settings			
✓		DIMMER_INFO	0x0340		
✓	✓	MINIMUM_LEVEL	0x0341		
✓	✓	MAXIMUM_LEVEL	0x0342		
✓	✓	CURVE	0x0343		
✓		CURVE_DESCRIPTION	0x0344	* Support required only if CURVE is supported.	✓*
✓	✓	OUTPUT_RESPONSE_TIME	0x0345		
✓		OUTPUT_RESPONSE_TIME_ DESCRIPTION	0x0346	* Support required only if OUTPUT_RESPONSE_ TIME is supported.	✓*
✓	✓	MODULATION_FREQUENCY	0x0347		
✓		MODULATION_FREQUENCY_ DESCRIPTION	0x0348	* Support required only if MODULATION_ FREQUENCY is supported.	✓*
		Category – Power/Lamp Settings			
✓	✓	BURN_IN	0x0440		
		Category – Configuration			
✓	✓	LOCK_PIN	0x0640		
✓	✓	LOCK_STATE	0x0641		
✓		LOCK_STATE_DESCRIPTION	0x0642	* Support required only if LOCK_STATE is supported.	✓*
		Category – Control			
✓	✓	IDENTIFY_MODE	0x1040		
✓		PRESET_INFO	0x1041		
✓	✓	PRESET_STATUS	0x1042		
✓	✓	PRESET_MERGEMODE	0x1043	See Table A-3	
✓	✓	POWER_ON_SELF_TEST	0x1044		

Table A-2: Preset Programmed Defines

Preset Programmed Defines	Value	Comment
PRESET_NOT_PROGRAMMED	0x00	Preset Scene not programmed.
PRESET_PROGRAMMED	0x01	Preset Scene programmed
PRESET_PROGRAMMED_READ_ONLY	0x02	Preset Scene Read-Only, Factory Programmed

Table A-3: Merge Mode Defines

Merge Mode Defines	Value	Comment
MERGEMODE_DEFAULT	0x00	Preset overrides DMX512 default behavior as defined in E1.20 PRESET_PLAYBACK.
MERGEMODE_HTP	0x01	Highest Takes Precedence on slot by slot basis
MERGEMODE_LTP	0x02	Latest change takes precedence from Preset or DMX512 on a slot by slot basis
MERGEMODE _DMX_ONLY	0x03	DMX512 only, Preset ignored
MERGEMODE_OTHER	0xFF	Other (undefined) merge mode

== END ==