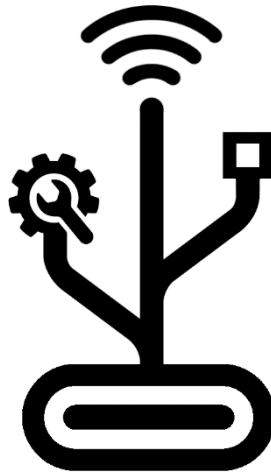


Universal Connect Module standard

Version 1.20



Revision History

Date	Document Revision	Standard Revision	Description
17/11/2025	1	1.20	Initial Public Release

Contents

Revision History	2
1. Introduction	5
1.1. What is the Universal Connect Module standard?	5
1.2. Universal Connect Module types	5
1.3. Lighting Fixture types	5
1.4. Open Standard	5
2. Standard Versions	6
2.1. Version 1.10.....	6
2.2. Version 1.20.....	6
3. Principle of Operation	7
3.1. Mechanical Interface.....	7
3.2. Electrical Interface.....	7
3.3. Protocol Interface.....	7
3.4. Module Configuration and Status Readout.....	7
3.5. RDM Pass Through	8
3.6. DMX Data	8
4. Mechanical Interface.....	9
4.1. Fixture Side.....	9
4.2. Module Side	9
5. Electrical Interface.....	11
6. Protocol Interface.....	12
6.1. Communications Device Class.....	12
6.2. Serial Line Internet Protocol.....	12
6.3. UCM Packet Structure	13
7. UCM DMX/RDM Frame	14
7.1. UCM DMX Frame Example	14
7.2. UCM RDM Frame Example (Module to Slot).....	14
7.3. UCM RDM Frame Example (Slot to Module).....	14
8. UCM RDM PID Set Frame	16
9. UCM RDM Sensor Set Frame	17
10. UCM Version Frame	18
10.1. Module Version 1.10 and Fixture Version 1.10	18

10.2. Module Version 1.10 and Fixture Version 1.20	18
10.3. Module Version 1.20 and Fixture Version 1.10	18
10.4. Module Version 1.20 and Fixture Version 1.20	18
11. UCM RDM PID Define Frame	19
12. UCM RDM PID Define Enum Text Frame	21
13. UCM RDM Sensor Define Frame	22
14. RDM Parameters and Sensors.....	24
14.1. Mandatory non-hidden UCM RDM Parameters.....	24
14.2. Generic UCM RDM Parameters.....	24
14.3. Module-Specific UCM RDM Parameters	25
14.4. RDM GET/SET Logic	26
14.5. Generic UCM RDM Sensors.....	26
14.6. Module-Specific UCM RDM Sensors	27
14.7. UCM RDM Parameter and Sensor Initialization	27
15. Step-by-Step Initialization Guide.....	28
15.1. USB Enumeration	28
15.2. RDM Discovery	28
15.3. UCM Version Check.....	29
15.4. UCM Initialization.....	29
15.5. Initialize Module-Specific UCM RDM Parameters.....	31
15.6. Define generic UCM RDM Parameters	33
15.7. Synchronize Module-Specific Parameters.....	34
15.8. Initialize Module-Specific UCM RDM Sensors.....	34
15.9. Define Generic UCM RDM Sensors	35
15.10. Synchronize Module-Specific Sensors.....	36
15.11. Additional Initialization Steps.....	36
16. Continuous Operation Guide	37
16.1. Send DMX Frames to the fixture when available	37
16.2. Pass on RDM frames	37
16.3. Perform RDM Get requests for module-specific parameters	37
16.4. Update read-only module-specific parameters via UCM RDM PID Set	38
16.5. Update module-specific sensors via UCM RDM Sensor Set.....	38

1. Introduction

1.1. What is the Universal Connect Module standard?

The Universal Connect Module standard defines a standard mechanical form-factor, electrical interface and communication protocol for accessory plug-in modules for lighting fixtures. Enabling a wide range of different plug-in modules to be compatible with a wide range of different (stage) lighting fixtures.

1.2. Universal Connect Module types

Universal Connect Modules can add a wide variety of functionalities to (stage) lighting fixtures: wireless connectivity, (proprietary) wired network connectivity, show recording, long range wireless connectivity and more.

1.3. Lighting Fixture types

Any type of (stage) lighting fixture can be fitted with a Universal Connect Module slot: moving lights, static lights, effect lights and more. The Universal Connect Module standard defines an IP65-rated module and slot, enabling outdoor-rated lighting fixtures to use this standard as well.

1.4. Open Standard

While the Universal Connect Module standard was originally developed through a collaboration between City Theatrical and Martin Professional, it is meant to be an open standard. Any module or fixture manufacturer can use the Universal Connect Module standard freely.

Suggestions to extend or improve the standard are welcome at martinsoftware@harman.com and support@citytheatrical.com.

2. Standard Versions

Below chapter outlines the features that were added or changed in each version of the Universal Connect Module standard.

2.1. Version 1.10

Initial version of the Universal Connect Module standard. Does not include the UCM RDM PID Define, UCM RDM PID Define Enum Text and RDM Sensor Define Frames.

Fixtures that comply with version 1.10 of the standard can only communicate with Modules for which they know the module-specific RDM parameters and RDM sensors (as-in: communicate with Modules for which they have hardcoded definitions in their firmware).

Modules that comply with version 1.10 of the standard can only communicate with Fixtures that know their module-specific RDM parameters and RDM sensors.

2.2. Version 1.20

This version adds support for the UCM RDM PID Define, UCM RDM PID Define Enum Text and RDM Sensor Define Frames. And the requirement for a Fixture to offer Generic UCM RDM parameters and sensors.

These enable a Universal Connect Module to configure these Generic UCM RDM parameters and RDM sensors for its own usage. This enables Modules to work with Fixtures without requiring module-specific firmware work to be done on the fixture side.

Fixtures that comply with version 1.20 of the standard will offer generic RDM parameters and RDM sensors and allow a Module to configure these. Fixtures that comply with version 1.20 of the standard shall also support the module-specific RDM parameters and RDM sensors defined in this document, as this enables them to support version 1.10 Modules as well.

Modules that comply with version 1.10 of the standard will be able to configure generic RDM parameters and RDM sensors offered by the Fixtures. Modules that comply with version 1.20 of the standard shall also be able to communicate with Fixtures without generic RDM parameters and RDM sensors, if the Fixture offers the module-specific RDM parameters and RDM sensors needed by the Module.

3. Principle of Operation

Below chapter provides a high-level description of the Universal Connect Module standard and hence provides a good starting point. A detailed description of each part is covered in the following chapters.

3.1. Mechanical Interface

The Universal Connect Module standard defines a mechanical form-factor for a “module” and a “slot”. The interface between module and slot is sealed, enabling use of this interface in outdoor-rated applications. The standard also defines an antenna-clip mounting point near the slot, so that modules can be supplied with an antenna-clip which matches the used antenna (if relevant).

3.2. Electrical Interface

A standard USB Type-C connector is used between module and slot. Pinout and signals also follow the USB Type-C standard for a full-speed USB 2.0 interface. Universal Connect Modules are also powered through the USB Type-C connector at 5 Vdc up to 100 mA.

3.3. Protocol Interface

Universal Connect Modules enumerate as a USB communications device class (USB CDC) device. Modules may also enumerate as a device in a different USB class when they detect they are not connected to a compatible lighting fixture (for example: enumerate as a USB mass storage class (USB MSC) device, when not connected to a fixture, to support firmware upload from a standard computer).

Communication between fixture and module is then wrapped in the serial line internet protocol (SLIP) before sending it over the virtual serial port of the USB CDC.

A very simple UCM protocol is then used to transmit RDM and DMX packets over the serial line internet protocol.

3.4. Module Configuration and Status Readout

Universal Connect Modules can be configured through the fixture into which they are plugged into. A Universal Connect Module will act as a RDM controller and read specific RDM parameters from the fixture into which it is plugged. This enables the fixture to change all settings of the Universal Connect Module (for example: radio channel, output power, ...).

Universal Connect Modules can also feedback their status to the fixture into which they are plugged into. A Universal Connect Module will override specific RDM sensors of the fixture into which it is plugged. This enables the fixture to display the module status via sensor readout (for example: signal quality, signal strength, ...).

To maintain changes that happen at the UI of the Fixture, the Module will poll the Fixture for any changes in these settings periodically. Therefore, when settings relevant to the UCM Module are updated on the UI of the Fixture, the Fixture must make these changes reflected in the RDM message count.

3.5. RDM Pass Through

When a Universal Connect Module can connect with an RDM controller “upstream” (for example: Universal Connect Module is providing wireless DMX connectivity or network connectivity), it should act as an RDM pass-through device. The Module will be passive and assume the RDM UID of the Fixture. Any RDM command directed to the UID of the Fixture will be forwarded to the Fixture and the response will be forwarded back to the RDM controller. Enabling the link to be fully transparent for RDM communication as well, without breaking any part of the RDM standard.

Settings can be changed either by the user via the UI of the fixture, or through RDM directly to the UCM Module. To keep settings in sync between the Module and the Fixture, the Module will need to “sniff” the RDM traffic between an RDM controller and the Fixture.

To maintain changes that happen through RDM, the UCM Module will listen to the RDM traffic and set the values as necessary upon reception. This will allow immediate changes, as the polling nature of syncing the settings may cause a noticeable delay. The RDM commands will be forwarded to the Fixture.

The UCM Module will need to differentiate between messages sent by the RDM controller and messages sent by itself; and not forward responses back to the controller on messages that were not generated by the controller. This will be accomplished by using the Module’s RDM UID as the source UID for the messages.

Above is not relevant when the Universal Connect Module does not provide an RDM “uplink” to an “upstream” RDM controller (for example: Universal Connect Module is providing a local DMX playback function).

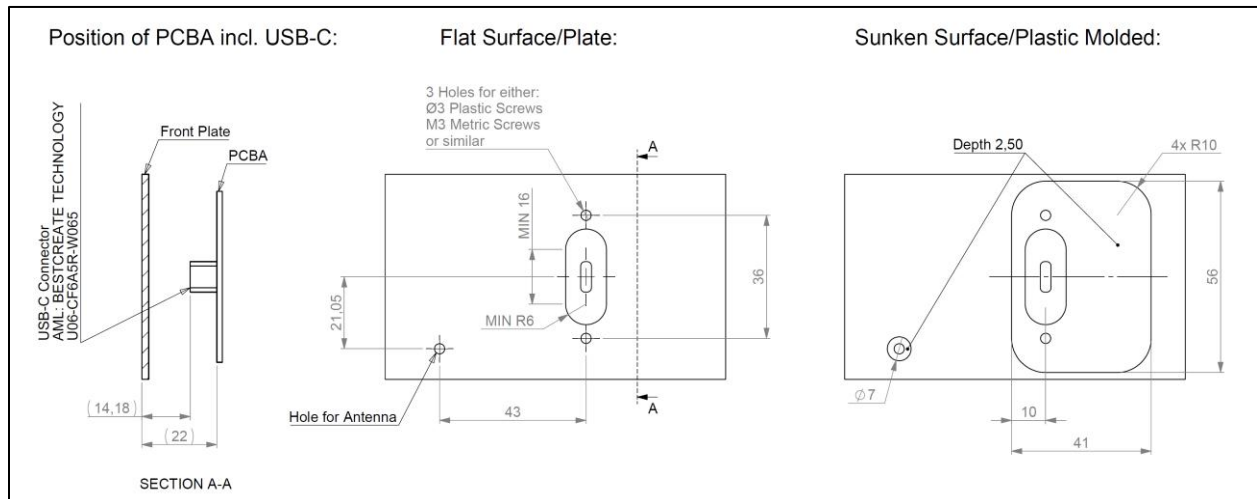
3.6. DMX Data

Full DMX frames are transmitted between module and fixture, wrapped in the protocols described above.

4. Mechanical Interface

4.1. Fixture Side

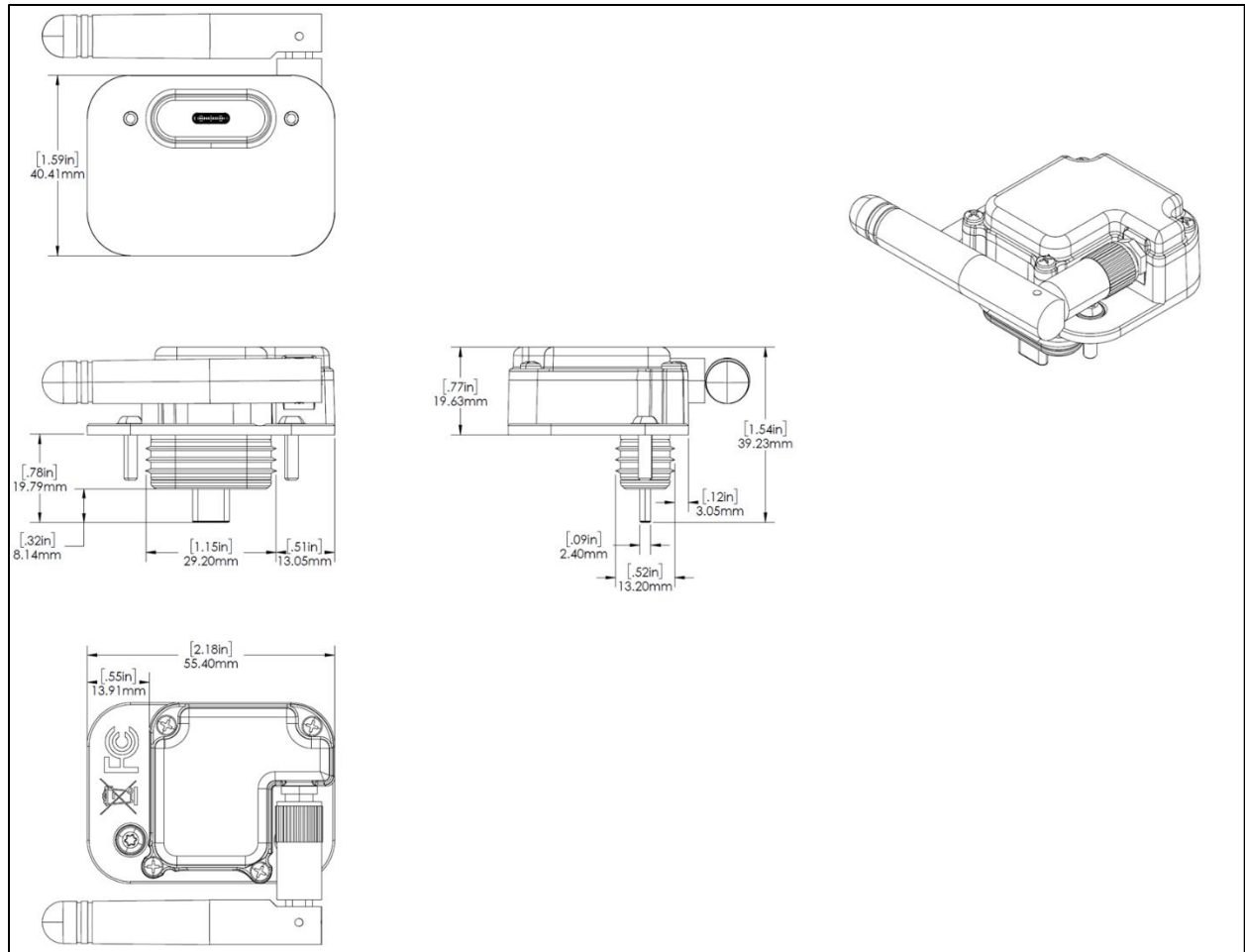
A Universal Connect Module Slot shall have dimensions as shown on below drawing (2D and 3D drawings also attached to this standard).



The hole for antenna enables an antenna-clip (supplied with a Universal Connect Module that features an antenna) to be fixated permanently to the lighting fixture housing. The screw to mount the antenna-clip is typically supplied with the fixture, to allow for different screw types to be used on different lighting fixture types (plastic, metal, threaded, ...).

4.2. Module Side

A Universal Connect Module shall have dimensions as shown on below drawing (2D and 3D drawings also attached to this standard).



A module is allowed to be higher than specified on the drawing but shall not exceed the specified width and length as that could compromise compatibility with existing lighting fixtures.

5. Electrical Interface

The connection between a Universal Connect Module and Universal Connect Module Slot is made using a standard (IP65-rated) USB Type-C connector.

The female USB Type-C connector on a UCM-compatible Fixture shall be wired for an USB 2.0 interface.

USB Type-C Connector	Usage
A1, A12, B1, B12	GND
A4, A9, B4, B9	Vbus
A5	56k Ω pull up to Vbus
B5	56k Ω pull up to Vbus
A7, B7	D-
A6, B6	D+
Shell	Shield

The male USB Type-C connector on a UCM Module shall be wired for an USB 2.0 interface.

USB Type-C Connector	Usage
A1, A12, B1, B12	GND
A4, A9, B4, B9	Vbus
A5	5.1k Ω pull down to GND
B5	Open
A7, B7	D-
A6, B6	D+
Shell	Shield

A Universal Connect Module Slot shall supply up to 100 mA at 5 Vdc via the Vbus pins, without negotiation.

A Universal Connect Module shall draw maximum 100 mA at 5 Vdc via the Vbus pins.

Both UCM Module and UCM Slot shall be designed to withstand hot-plug events (plug or unplug UCM Module while Fixture is powered on).

A UCM Slot shall be a USB Host. A UCM Module shall be a USB Device.

6. Protocol Interface

Universal Connect Module communication shall follow the USB 2.0 Full Speed standard.

6.1. Communications Device Class

A Universal Connect Module shall present itself as a USB communications device class (USB CDC) device, exposing a virtual serial port. All communication between the Module and the Fixture shall then be transmitted over this serial port.

6.2. Serial Line Internet Protocol

All data sent from the Fixture to the Module and from the Module to the Fixture shall be encapsulated in the Serial Line Internet Protocol (SLIP, following RFC 1015), using following special characters:

Hex Value	Dec Value	Abbreviation	Description
0xC0	192	END	Frame End
0xDB	219	ESC	Frame Escape
0xDC	220	ESC_END	Transposed Frame End
0xDD	221	ESC_ESC	Transposed Frame Escape

SLIP modifies a standard packet by:

1. Appending a special “END” byte to it, which distinguished the packet’s boundary.
2. If the END byte occurs in the data to be sent, the two-byte sequence ESC, ESC_END, is sent instead.
3. If the ESC byte occurs in the data, the two-byte sequence, ESC, ESC_ESC is sent.

Further information about the protocol can be found at <https://www.rfc-editor.org/rfc/rfc1055.txt>.

Example code on how to implement can be found below:

```
#define END          (0xC0)
#define ESC          (0xDB)
#define ESC_END      (0xDC)
#define ESC_ESC      (0xDD)

uint8_t srcBuffer[];    // Replace with your source buffer
uint8_t destBuffer[];   // Replace with your destination buffer

int length = 0;         // Length of final packet

for ( int i = 0; i < sizeof(srcBuffer); i++ )
{
    // Check each character for an escape sequence
    switch ( srcBuffer[i] )
    {
        case END:
            destBuffer[ length++ ] = ESC;
            destBuffer[ length++ ] = ESC_END;
            break;
        case ESC:
            destBuffer[ length++ ] = ESC;
            destBuffer[ length++ ] = ESC_ESC;
            break;
        default:
            destBuffer[ length++ ] = srcBuffer[i];
            break;
    }
}
// Append last byte
destBuffer[ length++ ] = END;
```

6.3. UCM Packet Structure

All packets are formatted as follows, before wrapping them in SLIP, and sending them using USB CDC, from Module to Fixture or vice-versa.

Bytes	Field	Notes
1	UCM Start Code	Always 0x00
1	UCM Command	See table below
2	UCM Length	Length of payload, excluding any bytes added by SLIP encoding
2	UCM Command Data	See table below
1 - 513	UCM Payload	Payload of the command

The UCM Command field can have following values:

Value	Description	Usage of UCM Command Data Field	Notes
0x01	DMX/RDM Frame	DMX Universe	Standard DMX/RDM packet.
0x02	RDM PID Set	RDM PID	Write an RDM PID that can only be read using standard RDM packet.
0x03	RDM Sensor Set	RDM Sensor Number	Set a RDM sensor value that can only be read using standard RDM packet.
0x04	Version	UCM Version Number	Send UCM Version between module and fixture and vice-versa.
0x05	RDM PID Define	RDM PID	Define a RDM PID on the Fixture to be used by the Module
0x06	RDM PID Define Enum Text	RDM PID	Define RDM PID Enum Values on the Fixture to be used by the Module
0x07	RDM Sensor Define	RDM Sensor Number	Define a Sensor on the Fixture to be used by the Module

All multi-byte values are transmitted in Big Endian byte order.

7. UCM DMX/RDM Frame

The payload of a UCM DMX/RDM Frame always starts with a Start Code, following ANSI E1.11. The message following should conform to the protocols that correlate to those start codes.

Start Codes are reserved as follows:

Value	Description	Protocol
0x00	DMX512	DMX Frame following ANSI 1.11.
0xCC	RDM	RDM Frame following ANSI 1.20.
0x43	City Theatrical Proprietary	(Reserved for future use by City Theatrical)
Other	Available	Other module manufacturers can reserve a proprietary start code, which will then be added to this standard.

7.1. UCM DMX Frame Example

A typical UCM DMX Frame sent from the Module to the Fixture has following structure:

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	0x0201 (513 bytes)
2	UCM Command Data	0x0111 (Universe 273)
513	UCM Payload	Start Code 0x00 + 512 bytes of DMX slot data

(please be aware that this table omits any characters added by the SLIP encoding)

7.2. UCM RDM Frame Example (Module to Slot)

Within UCM, the Module shall always act as an RDM Controller, and the Fixture shall always act as an RDM Device. A typical UCM RDM Frame sent from the Module to the Fixture has following structure (example: GET_DEVICE_INFO):

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	0x0018 (24 bytes)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
24	UCM Payload	Start Code 0xCC + RDM Packet from Sub-START Code till Parameter Data Length

(please be aware that this table omits any characters added by the SLIP encoding)

7.3. UCM RDM Frame Example (Slot to Module)

A typical UCM RDM Frame sent from the Fixture to the Module has following structure (example: GET_DEVICE_INFO RESPONSE):

Bytes	Field	Value
1	UCM Start Code	0x00

1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	0x002B (43 bytes)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
43	UCM Payload	Start Code 0xCC + RDM Packet from Sub-START Code till Sensor Count

(please be aware that this table omits any characters added by the SLIP encoding)

8. UCM RDM PID Set Frame

A UCM compatible Fixture has a few RDM PIDs to indicate Manufacturer, Model and Firmware Version of the UCM Module (see chapter RDM Parameters and Sensors). Given that these PIDs are read only, the UCM Module cannot set (write) these.

A UCM RDM PID Set Frame allows these PIDs to be set (written), and has following structure (example: write value 0x77 to PID 0x8005):

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x02 (RDM PID Set)
2	UCM Length	0x0001 (1 byte)
2	UCM Command Data	0x8005 (RDM PID 0x8005)
1	UCM Payload	0x77 (Value 0x77)

(please be aware that this table omits any characters added by the SLIP encoding)

These UCM RDM PID Set Frames are sent strictly from the Module to the Fixture. The Fixture should not respond. The Module can obtain confirmation that the PID was written correctly by sending a regular UCM RDM Get.

The module should only attempt to set RDM Manufacturer-specific Parameter IDs that it has identified to be used for UCM purposes. If the module attempts to set other RDM PIDs, the host fixture is to ignore these messages.

The module can also use UCM RDM PID Set to initialize an RDM PID it defined via UCM RDM PID Define. This can be used by the module to initialize an RDM PID to a specific value after creating it, or to restore an RDM PID to the value it saved locally before a power cycle.

9. UCM RDM Sensor Set Frame

A UCM compatible Fixture can have a few RDM Sensors to monitor module parameters such as Signal Quality and Signal Strength (see chapter RDM Parameters and Sensors). Given that RDM Sensors are read only, the UCM Module cannot set (write) these.

A UCM RDM Sensor Set Frame allows these Sensors to be set (written), and has following structure (example: write value 0x55AA to Sensor 10):

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x03 (RDM Sensor Set)
2	UCM Length	0x0002 (2 bytes)
2	UCM Command Data	0x000A (RDM Sensor 0x000A)
2	UCM Payload	0x55AA (Value 0x55AA)

(please be aware that this table omits any characters added by the SLIP encoding)

These UCM RDM Sensor Set Frames are sent strictly from the Module to the Fixture. The Fixture should not respond. The Module can obtain confirmation that the Sensor was written correctly by sending a regular UCM RDM Sensor Get.

The module should only attempt to set RDM Sensor Numbers that it has identified to be used for UCM purposes. If the module attempts to set other RDM Sensor Numbers, the host fixture is to ignore these messages.

RDM Sensor Number shall be in the range 0x0000 – 0x00FE, as define by ANSI 1.20.

10. UCM Version Frame

To make sure that UCM Module and UCM-compatible Fixture are using compatible versions of the UCM standard, they shall exchange Version numbers via a dedicated packet. The Module shall send this frame, and the Fixture shall respond to it with its own Version.

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x04 (Version)
2	UCM Length	0x0000 (0 bytes)
2	UCM Command Data	0x010A (Version 1.10)
0	UCM Payload	<blank>

(please be aware that this table omits any characters added by the SLIP encoding)

10.1. Module Version 1.10 and Fixture Version 1.10

Module and Fixture shall communicate via hardcoded RDM Parameters and Sensors only (as defined in “Module-Specific RDM Parameters” and “Module-Specific RDM Sensors”). UCM RDM PID Define, UCM RDM PID Define Enum Text and RDM Sensor Define Frames are not used.

10.2. Module Version 1.10 and Fixture Version 1.20

Module and Fixture shall communicate via hardcoded RDM Parameters and Sensors only (as defined in “Module-Specific RDM Parameters” and “Module-Specific RDM Sensors”). UCM RDM PID Define, UCM RDM PID Define Enum Text and RDM Sensor Define Frames are not used.

10.3. Module Version 1.20 and Fixture Version 1.10

Module and Fixture shall communicate via hardcoded RDM Parameters and Sensors only (as defined in “Module-Specific RDM Parameters” and “Module-Specific RDM Sensors”). UCM RDM PID Define, UCM RDM PID Define Enum Text and RDM Sensor Define Frames are not used.

10.4. Module Version 1.20 and Fixture Version 1.20

Module and Fixture can communicate via hardcoded RDM Parameters and Sensors, if the Fixture offers all UCM RDM Parameters and Sensors required by the Module.

Module and Fixture can communicate via Generic UCM RDM PIDs and RDM Sensors, which are first defined by the Module, using UCM RDM PID Define, UCM RDM PID Define Enum Text and RDM Sensor Define Frames.

11. UCM RDM PID Define Frame

A UCM-compatible Fixture shall offer a range of undefined/generic RDM PIDs which a UCM Module can define for its own utilization. Once a UCM Module has defined such an RDM PID, the Fixture will add it to its Supported Parameters and local display (using the Description field).

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x05 (RDM PID Define)
2	UCM Length	0x0012-0x0032 (18-50 bytes)
2	UCM Command Data	RDM PID to be defined
18-50	UCM Payload	PDL Size (1 byte per E1.20 PARAMETER_DESCRIPTION) Data Type (1 byte per E1.20 PARAMETER_DESCRIPTION) Command Class (1 byte per E1.20 PARAMETER_DESCRIPTION) Type (1 byte per E1.20 PARAMETER_DESCRIPTION) Unit (1 byte per E1.20 PARAMETER_DESCRIPTION) Prefix (1 byte per E1.20 PARAMETER_DESCRIPTION) Min Valid Value (4 bytes per E1.20 PARAMETER_DESCRIPTION) Max Valid Value (4 bytes per E1.20 PARAMETER_DESCRIPTION) Default Value (4 bytes per E1.20 PARAMETER_DESCRIPTION) Description (0-32 bytes per E1.20 PARAMETER_DESCRIPTION)

(please be aware that this table omits any characters added by the SLIP encoding)

Example of an UCM RDM PID Define (define RDM PID 0x80FF to be used for setting the Module “Speed”):

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x05 (RDM PID Define)
2	UCM Length	0x0018 (24 bytes)
2	UCM Command Data	0x80FF (RDM PID 0x80FF)
24	UCM Payload	0x01 (PDL: 1 byte) 0x03 (Data Type: DS_UNSIGNED_BYTE) 0x03 (Command Class: CC_GET_SET) 0x00 (Type: N/A) 0x00 (Unit: UNITS_NONE) 0x00 (Prefix: PREFIX_NONE) 0x00000000 (Min Valid Value: 0) 0x00000002 (Max Valid Value: 2) 0x00000001 (Default Value: 1) 0x53 0x70 0x65 0x65 0x64 0x00 (Description: Speed)

(please be aware that this table omits any characters added by the SLIP encoding)

These UCM RDM PID Define Frames are sent strictly from the Module to the Fixture. The Fixture should not respond. The Module can obtain confirmation that the PID was defined correctly by sending regular UCM RDM commands.

The Module should only attempt to define RDM Manufacturer-specific Parameter IDs that the Fixture has offered to be used for UCM purposes. If the module attempts to define other RDM PIDs, the host fixture is to ignore these messages.

After using UCM RDM PID Define to “create” an RDM PID on the Fixture, the Module shall use UCM RDM PID Set to initialize it to a starting value. This can be the default state of that RDM PID, or the state saved locally inside the Module from previous use.

12. UCM RDM PID Define Enum Text Frame

A UCM Module can inform the Fixture about the meaning of each enum value for a UCM-specific RDM PID. This enables the Fixture to show meaningful menu entries on its local display.

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x06 (RDM PID Define Enum Text)
2	UCM Length	0x0004-0x0024 (4-36 bytes)
2	UCM Command Data	RDM PID to define an enum value on
4-36	UCM Payload	Enum Value (4 bytes) Enum Description (ASCII Text Field of variable size 0-32 chars)

(please be aware that this table omits any characters added by the SLIP encoding)

Example of an UCM RDM PID Define Enum Text (define value 1 on RDM PID 0x80FF to be “Fast”):

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x06 (RDM PID Define Enum Text)
2	UCM Length	0x0009 (9 bytes)
2	UCM Command Data	0x80FF (RDM PID 0x80FF)
9	UCM Payload	0x00000001 (Enum Value: 1) 0x46 0x61 0x73 0x74 0x00 (Enum Description: Fast)

(please be aware that this table omits any characters added by the SLIP encoding)

These UCM RDM PID Define Enum Text Frames are sent strictly from the Module to the Fixture. The Fixture should not respond.

The Module should only attempt to define enum values on RDM Manufacturer-specific Parameter IDs that the Fixture has offered to be used for UCM purposes. If the module attempts to define enum values on other RDM PIDs, the host fixture is to ignore these messages.

A Module shall not attempt to create more than 1024 bytes worth of Enum Description inside the Fixture (as this would put unrealistic requirements on Fixture memory allocation). So, the sum of the length of all Enum Description fields sent (to all RDM PIDs and all Enum Values) shall not exceed 1024 bytes.

13. UCM RDM Sensor Define Frame

A UCM-compatible Fixture shall offer a range of undefined/generic RDM Sensor Numbers which a UCM Module can define for its own utilization. Once a UCM Module has defined such an RDM Sensor Number, the Fixture will add it to its Sensor Definition and local display (using the Description field). This enables the Fixture to display the status and other feedback values from the Module.

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x07 (RDM Sensor Define)
2	UCM Length	0x000C-0x002C (12-44 bytes)
2	UCM Command Data	RDM Sensor Number to be defined
12-44	UCM Payload	Type (1 byte per E1.20 SENSOR_DEFINITION) Unit (1 byte per E1.20 SENSOR_DEFINITION) Prefix (1 byte per E1.20 SENSOR_DEFINITION) Range Minimum Value (2 bytes per E1.20 SENSOR_DEFINITION) Range Maximum Value (2 bytes per E1.20 SENSOR_DEFINITION) Normal Minimum Value (2 bytes per E1.20 SENSOR_DEFINITION) Normal Maximum Value (2 bytes per E1.20 SENSOR_DEFINITION) Recorded Value Support (1 byte per E1.20 SENSOR_DEFINITION) Description (0-32 bytes per E1.20 PARAMETER_DESCRIPTION)

(please be aware that this table omits any characters added by the SLIP encoding)

Example of an UCM RDM Sensor Define (define Sensor Number 0x00F0 to be used for reading the Module's "Signal Strength"):

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x07 (RDM Sensor Define)
2	UCM Length	0x001C (28 bytes)
2	UCM Command Data	0x00F0 (Sensor Number 0x00F0)
28	UCM Payload	0x05 (Type: SENS_POWER) 0x00 (Unit: UNITS_NONE) 0x00 (Prefix: PREFIX_NONE) 0xFFEC (Range Minimum Value: -20) 0x0078 (Range Maximum Value: 120) 0x0000 (Normal Minimum Value: 0) 0x0064 (Normal Maximum Value: 100) 0x00 (Recorded Value Support: None) 0x53 0x69 0x67 0x6e 0x61 0x6c 0x20 0x53 0x74 0x72 0x65 0x6e 0x67 0x74 0x68 0x00 (Description: Signal Strength)

(please be aware that this table omits any characters added by the SLIP encoding)

These UCM RDM Sensor Define Frames are sent strictly from the Module to the Fixture. The Fixture should not respond. The Module can obtain confirmation that the Sensor was defined correctly by sending regular UCM RDM commands.

The Module should only attempt to define RDM Sensor Numbers that the Fixture has offered to be used for UCM purposes. If the module attempts to define other RDM Sensor Numbers, the host fixture is to ignore these messages.

After using UCM RDM Sensor Define to “create” a Sensor on the Fixture, the Module shall use UCM RDM Sensor Set to initialize it to its starting value.

14. RDM Parameters and Sensors

RDM PIDs are used between a UCM Module and UCM-compatible Fixture to discover and configure the UCM Module. RDM Sensors are used by a UCM-compatible Fixture to read out the status of the UCM Module.

During initialization, a UCM Module will verify if the UCM-compatible Fixture offers the PIDs and Sensors required for the Module to communicate. The process works as follows:

1. Module queries the Fixture for its supported parameters via RDM SUPPORTED_PARAMETERS.
2. Module queries the Fixture for the description of each of the supported parameters via RDM PARAMETER_DESCRIPTION.
3. Module queries the Fixture for the description of each of the RDM Sensors via RDM SENSOR_DEFINITION.
4. Once the Module has found the needed parameters and sensors, it can communicate with the fixture via the found RDM PIDs and RDM Sensor Numbers.
5. If the Module cannot find the parameters and sensors it needs, it will search for generic UCM RDM PIDs and Sensors. And then define these for its use via UCM RDM PID Define Frame, UCM RDM PID Enum Text Frame and UCM RDM Sensor Define Frame.

14.1. Mandatory non-hidden UCM RDM Parameters

A UCM-compatible Fixture shall always support the 3 RDM PIDs listed below. These enable the UCM Module to confirm that the Fixture supports the UCM standard, and enable the UCM Module to write its Manufacturer, Model and Firmware Version information to the Fixture via UCM RDM PID Set Frames.

This then enables the Fixture to show the Manufacturer, Model and Firmware Version of the inserted UCM Module on its local display.

The Fixture manufacturer can freely choose which (3) RDM PID numbers to use for these, as the UCM Module will “find” them by querying the text description.

Text Description	Class	Data Type	Command Class	Usage
UCM MANUFACTURER	General	DS_ASCII	GET	Module Manufacturer Description
UCM MODEL	General	DS_ASCII	GET	Module Model Description
UCM FIRMWARE VERSION	General	DS_ASCII	GET	Module Firmware Version Description

(please be aware that the text description is not case-sensitive and UCM Modules shall “accept” RDM PIDs in lower, upper and mixed case)

14.2. Generic UCM RDM Parameters

A UCM-compatible Fixture shall offer Generic UCM RDM PIDs. These can then be configured by the UCM Module for its own use via UCM RDM PID Define and UCM RDM PID Define Enum Text Frames.

Once one or more of these Generic UCM RDM PIDs are configured by the UCM Module, the Fixture shall add them as entries to its local display to enable the user to configure the module and read out information.

A UCM-compatible Fixture shall offer 16 of these Generic UCM RDM PIDs, to support UCM Modules with up to 16 parameters.

A UCM-compatible Fixture is allowed to hide these PIDs from its SUPPORTED_PARAMETERS till UCM MANUFACTURER, UCM MODEL and UCM FIRMWARE VERSION have been written by the UCM Module. This enables the Fixture to maintain a clean list of RDM PIDs when no UCM Module is plugged into it.

The Fixture manufacturer can freely choose which (16) RDM PID numbers to use for these, as the UCM Module will find them by querying the text description.

Text Description	Class	Data Type	Command Class	Usage
UCM GENERIC	General	DS_ASCII	GET/SET	Freely defined by UCM Module

(please be aware that the text description is not case-sensitive and UCM Modules shall “accept” RDM PIDs in lower, upper and mixed case)

14.3. Module-Specific UCM RDM Parameters

A UCM-compatible Fixture can also implement RDM PIDs for specific UCM Modules directly, as this enables the Fixture to communicate with the UCM Module without the UCM Module first needing to configure the Generic UCM RDM PIDs.

Below table lists all currently known module-specific RDM parameters. UCM Module manufacturers are encouraged to publish their module-specific Parameters, so that they can be added to this document, enabling fixture manufacturers to make their fixtures support these parameters.

A UCM-compatible Fixture is allowed to hide these PIDs from its SUPPORTED_PARAMETERS till UCM MANUFACTURER, UCM MODEL and UCM FIRMWARE VERSION have been written by the UCM Module. This enables the Fixture to maintain a clean list of RDM PIDs when no UCM Module is plugged into it.

The Fixture manufacturer can freely choose which (5) RDM PID numbers to use for these, as the UCM Module will find them by querying the text description.

Text Description	Class	Data Type	Command Class	Usage
CT SHOW ID	Module-Specific	DS_UNSIGNED_WORD	GET/SET	Multiverse SHoW ID
CT SHOW KEY	Module-Specific	DS_UNSIGNED_WORD	GET/SET	Multiverse Show Key
CT OUTPUT POWER	Module-Specific	DS_UNSIGNED_BYTE	GET/SET	Multiverse Output Power
CT ANTENNA TYPE	Module-Specific	DS_UNSIGNED_BYTE	GET/SET	Multiverse Antenna Type

CT CONNECTION STATE	Module-Specific	DS_UNSIGNED_BYTE	GET	Multiverse Connection State
---------------------	-----------------	------------------	-----	-----------------------------

(please be aware that the text description is not case-sensitive and UCM Modules shall “accept” RDM PIDs in lower, upper and mixed case)

Module-Specific RDM Parameters with prefix “CT” are reserved by City Theatrical. Please consult City Theatrical Multiverse Connect Module documentation for the detailed description of these Parameter.

14.4. RDM GET/SET Logic

It is important to realize that the Command Class of a UCM RDM PID should be seen from a “fixture + module combined” viewpoint, as it would be seen by an RDM controller external to the “fixture + module combination”.

Examples:

- GET: Wireless Connection State → Can only be read by an external RDM controller
- SET: Wireless Pairing Start → Can only be written by an external RDM controller
- GET/SET: Wireless Radio Channel → Can be read and written by an external RDM controller

14.5. Generic UCM RDM Sensors

A UCM-compatible Fixture shall offer Generic UCM RDM Sensors. These can then be configured by the UCM Module for its own use via UCM RDM Sensor Define Frames.

Once one or more of these Generic UCM RDM Sensors are configured by the UCM Module, the Fixture shall add them as entries to its local display to enable the user to read out module status and information.

A UCM-compatible Fixture shall offer 16 of these Generic UCM RDM Sensors, to support UCM Modules with up to 16 sensors.

A UCM-compatible Fixture is allowed to hide these Sensor Numbers from its DEVICE_INFO till UCM MANUFACTURER, UCM MODEL and UCM FIRMWARE VERSION have been written by the UCM Module. This enables the Fixture to maintain a clean list of Sensor Numbers when no UCM Module is plugged into it.

The Fixture manufacturer can freely choose which (16) RDM Sensor numbers to use for these, as the UCM Module will find them by querying the text description.

Text Description	Class	Type	Data Type	Prefix	Usage
UCM GENERIC	General	SENS_OTHER	UNITS_NONE	PREFIX_NONE	Freely defined by UCM Module

(please be aware that the text description is not case-sensitive and UCM Modules shall “accept” RDM Sensors in lower, upper and mixed case)

14.6. Module-Specific UCM RDM Sensors

A UCM-compatible Fixture can also implement RDM Sensors for specific UCM Modules directly, as this enables the Fixture to communicate with the UCM Module without the UCM Module first needing to configure the Generic UCM RDM Sensors.

Below table lists all currently known module-specific RDM sensors. UCM Module manufacturers are encouraged to publish their module-specific sensors, so that they can be added to this document, enabling fixture manufacturers to make their fixtures support these sensors.

A UCM-compatible Fixture is allowed to hide these Sensor Numbers from its DEVICE_INFO till UCM MANUFACTURER, UCM MODEL and UCM FIRMWARE VERSION have been written by the UCM Module. This enables the Fixture to maintain a clean list of Sensor Numbers when no UCM Module is plugged into it.

The Fixture manufacturer can freely choose which (2) RDM Sensor numbers to use for these, as the UCM Module will find them by querying the text description.

Text Description	Class	Type	Data Type	Prefix	Usage
CT SIGNAL QUALITY	Module-Specific	SENS_OTHER	UNITS_NONE	PREFIX_NONE	Multiverse Signal Quality
CT SIGNAL STRENGTH	Module-Specific	SENS_OTHER	UNITS_NONE	PREFIX_NONE	Multiverse Signal Strength

(please be aware that the text description is not case-sensitive and UCM Modules shall “accept” RDM Sensors in lower, upper and mixed case)

Module-Specific RDM Sensors with prefix “CT” are reserved by City Theatrical. Please consult City Theatrical Multiverse Connect Module documentation for the detailed description of these Sensors.

14.7. UCM RDM Parameter and Sensor Initialization

A UCM-compatible Fixture is not required to store the state of a UCM RDM Parameter or Sensor in non-volatile memory. It is hence the responsibility of the UCM Module to initialize all UCM RDM Parameters and Sensors during the initialization sequence (using RDM PID Set and RDM Sensor Set). A UCM Module can initialize these to a value it saved inside (for example: Wireless Radio Channel), a default value (for example: Wireless Connection State) or the current value (for example: Wireless Signal Strength).

15. Step-by-Step Initialization Guide

This chapter outlines all steps to be taken from UCM Module startup till continuous operation. Even though different models of UCM Modules will offer different handles (parameters), the basic process shall always be the same.

15.1. USB Enumeration

Once powered up (and/or plugged into a fixture) a UCM Module shall enumerate as a CDC (Communications Device Class) device and open a virtual serial port with the Fixture.

15.2. RDM Discovery

Once the serial port has been set up, the UCM Module will perform an RDM Discovery to determine if it has been plugged into a fixture (and obtain the RDM UID of that fixture).

Module to Fixture (RDM: DISC_UN_MUTE)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	0x001A (26 bytes)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
26	UCM Payload	RDM: DISC_UN_MUTE

(please be aware that this table omits any characters added by the SLIP encoding)

Module to Fixture (RDM: DISC_UNIQUE_BRANCH)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	0x0026 (38 bytes)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
38	UCM Payload	RDM: DISC_UNIQUE_BRANCH

(please be aware that this table omits any characters added by the SLIP encoding)

Fixture to Module (RDM: Unique Branch Response)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	0x001A (26 bytes)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
26	UCM Payload	RDM: Unique Branch Response

(please be aware that this table omits any characters added by the SLIP encoding)

If RDM discovery is successfully completed, the UCM Module will move on to the next step. Otherwise, the UCM Module may assume it is not plugged into a UCM-compatible fixture and re-enumerate under a

different USB Device Class (for example: switch into boot-loader mode or firmware upload mode, when plugged into a computer, as Mass Storage Device).

15.3. UCM Version Check

After RDM Discovery is completed, the Module and Fixture will exchange information about which revision of the UCM standard is being used. The Module initiates this transaction, and the following example demonstrates this with revision 1.10.

Module to Fixture (UCM Version)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x04 (Version)
2	UCM Length	0x0000 (0 bytes)
2	UCM Command Data	0x010A (Version 1.10)
0	UCM Payload	<blank>

(please be aware that this table omits any characters added by the SLIP encoding)

Fixture to Module (UCM Version)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x04 (Version)
2	UCM Length	0x0000 (0 bytes)
2	UCM Command Data	0x010A (Version 1.10)
0	UCM Payload	<blank>

(please be aware that this table omits any characters added by the SLIP encoding)

15.4. UCM Initialization

Once it has been confirmed that the UCM Module and Fixture are working with compatible versions of the UCM standard, the UCM Module will discover which RDM PIDs to use for set-only UCM RDM parameters and then inform the Fixture as to what type of UCM Module is plugged in.

The UCM Module will start by sending out a Get: SUPPORTED_PARAMETERS RDM message. It will use this list of PIDs to find the mandatory non-hidden UCM RDM parameters (UCM Manufacturer, UCM Model, UCM Firmware Version). It will not yet look for other UCM RDM parameters, as the fixture may not yet know what type of UCM Module was installed (and currently hide those parameters).

Module to Fixture (RDM: GET SUPPORTED_PARAMETERS)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	0x001A (26 bytes)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
26	UCM Payload	RDM: GET SUPPORTED_PARAMETERS

(please be aware that this table omits any characters added by the SLIP encoding)

Fixture to Module (RDM: SUPPORTED_PARAMETERS RESPONSE)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	26 + (# PIDs x 2)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
26 + (# PIDs x 2)	UCM Payload	RDM: SUPPORTED_PARAMETERS RESPONSE

(please be aware that this table omits any characters added by the SLIP encoding)

The UCM Module will now send out Get: PARAMETER_DESCRIPTION RDM messages for each of the reported PIDs until it finds PIDs with descriptions of "UCM Manufacturer", "UCM Model", and "UCM Firmware Version". These are not case sensitive. If the Module does not find these PIDs from the list of supported parameters, it will loop back through the list until it finds them.

Module to Fixture (RDM: GET_PARAMETER_DESCRIPTION)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	0x001C (28 bytes)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
28	UCM Payload	RDM: GET_PARAMETER_DESCRIPTION

(please be aware that this table omits any characters added by the SLIP encoding)

Fixture to Module (RDM: PARAMETER_DESCRIPTION RESPONSE)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	46-78 (depending on description length)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
46-78	UCM Payload	RDM: PARAMETER_DESCRIPTION RESPONSE

(please be aware that this table omits any characters added by the SLIP encoding)

Once the Module has discovered the UCM RDM parameters, it will set them using the UCM RDM PID Set command. The Fixture may use this information to determine how to respond accordingly. For example, if the fixture chooses to hide (from local display and RDM) the UCM parameters when there is no module connected, it will need to expose them at this point.

The fixture does not respond to these messages. The following three messages are sent in the order they appear.

Module to Fixture (UCM RDM PID Set: UCM Manufacturer)

Bytes	Field	Value
1	UCM Start Code	0x00

1	UCM Command	0x02 (RDM PID Set)
2	UCM Length	Length of payload string
2	UCM Command Data	UCM MANUFACTURER PID (as discovered above)
Length	UCM Payload	Zero-terminated string describing the Manufacturer of the UCM Module

(please be aware that this table omits any characters added by the SLIP encoding)

Module to Fixture (UCM RDM PID Set: UCM Model)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x02 (RDM PID Set)
2	UCM Length	Length of payload string
2	UCM Command Data	UCM MODEL PID (as discovered above)
Length	UCM Payload	Zero-terminated string describing the Model of the UCM Module

(please be aware that this table omits any characters added by the SLIP encoding)

Module to Fixture (UCM RDM PID Set: UCM Firmware Version)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x02 (RDM PID Set)
2	UCM Length	Length of payload string
2	UCM Command Data	UCM FIRMWARE VERSION PID (as discovered above)
Length	UCM Payload	Zero-terminated string describing the Firmware Version of the UCM Module

(please be aware that this table omits any characters added by the SLIP encoding)

Once this step has been completed, the Fixture is aware of the specific model of UCM Module that has been plugged into the fixture.

If the Fixture knows the specific model of UCM Module, it shall enable its module-specific RDM Parameters and Sensors and add them to its local display.

If the Fixture does not know the specific model of UCM Module, it shall enable the Generic UCM Parameters and Sensors, so that the UCM Module can configure these as needed (after which the Fixture will add them to its local display).

15.5. Initialize Module-Specific UCM RDM Parameters

The UCM Module will now discover the RDM PIDs where its specific parameters (for example: radio channel, pin-code, antenna type, ...) are defined. It will re-issue a SUPPORTED_PARAMETERS package, as the Fixture may have added parameters as part of the UCM Initialization.

Module to Fixture (RDM: GET SUPPORTED_PARAMETERS)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)

2	UCM Length	0x001A (26 bytes)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
26	UCM Payload	RDM: GET SUPPORTED_PARAMETERS

(please be aware that this table omits any characters added by the SLIP encoding)

Fixture to Module (RDM: SUPPORTED_PARAMETERS RESPONSE)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	26 + (# PIDs x 2)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
26 + (# PIDs x 2)	UCM Payload	RDM: SUPPORTED_PARAMETERS RESPONSE

(please be aware that this table omits any characters added by the SLIP encoding)

The UCM Module will now send out Get: PARAMETER_DESCRIPTION RDM messages for each PID until it finds all PIDs with descriptions matching the module-specific parameters. These are not case sensitive. Different types of UCM Modules will have different module-specific parameters.

Module to Fixture (RDM: GET PARAMETER_DESCRIPTION)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	0x001C (28 bytes)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
28	UCM Payload	RDM: GET PARAMETER_DESCRIPTION

(please be aware that this table omits any characters added by the SLIP encoding)

Fixture to Module (RDM: PARAMETER_DESCRIPTION RESPONSE)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	46-78 (depending on description length)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
46-78	UCM Payload	RDM: PARAMETER_DESCRIPTION RESPONSE

(please be aware that this table omits any characters added by the SLIP encoding)

If the UCM Module finds all its needed module-specific UCM RDM Parameters, it can proceed to step 14.7.

If the UCM Module does not find all its needed module-specific UCM RDM Parameters, it will need to configure some of the Generic UCM RDM Parameters for its own use in step 14.6.

15.6. Define generic UCM RDM Parameters

The UCM Module will now take one or multiple of the Generic UCM RDM PIDs and configure them for its own use. This then also informs the Fixture what parameters to show on its local display and how to represent them correctly.

Module to Fixture (UCM RDM PID Define)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x05 (RDM PID Define)
2	UCM Length	0x0012-0x0032 (18-50 bytes)
2	UCM Command Data	RDM PID to be defined
18-50	UCM Payload	PDL Size (1 byte per E1.20 PARAMETER_DESCRIPTION) Data Type (1 byte per E1.20 PARAMETER_DESCRIPTION) Command Class (1 byte per E1.20 PARAMETER_DESCRIPTION) Type (1 byte per E1.20 PARAMETER_DESCRIPTION) Unit (1 byte per E1.20 PARAMETER_DESCRIPTION) Prefix (1 byte per E1.20 PARAMETER_DESCRIPTION) Min Valid Value (4 bytes per E1.20 PARAMETER_DESCRIPTION) Max Valid Value (4 bytes per E1.20 PARAMETER_DESCRIPTION) Default Value (4 bytes per E1.20 PARAMETER_DESCRIPTION) Description (0-32 bytes per E1.20 PARAMETER_DESCRIPTION)

(please be aware that this table omits any characters added by the SLIP encoding)

The UCM Module will send UCM RDM PID Define to as many of the Generic UCM RDM PIDs as it needs for its own use (for example: Radio Channel, Antenna Type and Pincode). This then enables the Fixture to populate its local display with the needed parameters.

The UCM Module can also define the enum text for all/some of the RDM PIDs it just defined.

Module to Fixture (UCM RDM PID Enum Define)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x06 (RDM PID Define Enum Text)
2	UCM Length	0x0004-0x0024 (4-36 bytes)
2	UCM Command Data	RDM PID to define an enum value on
4-36	UCM Payload	Enum Value (4 bytes) Enum Description (ASCII Text Field of variable size 0-32 chars)

(please be aware that this table omits any characters added by the SLIP encoding)

For example, on the Antenna Type RDM PID, it can define which enum value corresponds with which type of antenna. This then enables the Fixture to show meaningful text in its local display.

15.7. Synchronize Module-Specific Parameters

Once the UCM Module has found (and/or defined) all its module-specific parameters, it will now use UCM RDM PID Set to initialize all of them to the correct value. This could be the default value, or the value stored in non-volatile memory inside the UCM Module.

Module to Fixture (UCM RDM PID Set)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x02 (RDM PID Set)
2	UCM Length	Variable (depending on parameter)
2	UCM Command Data	RDM PID
Length	UCM Payload	Depending on parameter

(please be aware that this table omits any characters added by the SLIP encoding)

15.8. Initialize Module-Specific UCM RDM Sensors

The UCM Module will now discover the RDM sensor numbers where its specific sensors (for example: signal strength, signal quality, ...) are defined. It will first issue a DEVICE_INFO package, to determine the number of RDM sensors.

Module to Fixture (RDM: GET DEVICE_INFO)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	0x001A (26 bytes)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
26	UCM Payload	RDM: GET DEVICE_INFO

(please be aware that this table omits any characters added by the SLIP encoding)

Fixture to Module (RDM: DEVICE_INFO RESPONSE)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	0x002D (45 bytes)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
45	UCM Payload	RDM: DEVICE_INFO RESPONSE

(please be aware that this table omits any characters added by the SLIP encoding)

Now the UCM Module will send out Get: SENSOR_DEFINITION RDM messages to each sensor number until it finds sensors that match its module-specific ones. These are not case sensitive. Different types of UCM Modules will have different module-specific sensors. For example: signal strength, signal quality, ...

Module to Fixture (RDM: GET SENSOR_DEFINITION)

Bytes	Field	Value
1	UCM Start Code	0x00

1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	0x001B (27 bytes)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
27	UCM Payload	RDM: GET SENSOR_DEFINITION

(please be aware that this table omits any characters added by the SLIP encoding)

Fixture to Module (RDM: SENSOR_DEFINITION RESPONSE)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	39-71 (depending on description length)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
39-71	UCM Payload	RDM: SENSOR_DEFINITION RESPONSE

(please be aware that this table omits any characters added by the SLIP encoding)

If the UCM Module finds all its needed module-specific UCM RDM Sensors, it can proceed to step 14.10.

If the UCM Module does not find all its needed module-specific UCM RDM Sensors, it will need to configure some of the Generic RDM Sensors for its own use in step 14.9.

15.9. Define Generic UCM RDM Sensors

The UCM Module will now take one or multiple of the Generic UCM RDM Sensors and configure them for its own use. This then also informs the Fixture what values to show on its local display and how to represent them correctly.

Module to Fixture (UCM RDM Sensor Define)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x07 (RDM Sensor Define)
2	UCM Length	0x000C-0x002C (12-44 bytes)
2	UCM Command Data	RDM Sensor Number to be defined
12-44	UCM Payload	Type (1 byte per E1.20 SENSOR_DEFINITION) Unit (1 byte per E1.20 SENSOR_DEFINITION) Prefix (1 byte per E1.20 SENSOR_DEFINITION) Range Minimum Value (2 bytes per E1.20 SENSOR_DEFINITION) Range Maximum Value (2 bytes per E1.20 SENSOR_DEFINITION) Normal Minimum Value (2 bytes per E1.20 SENSOR_DEFINITION) Normal Maximum Value (2 bytes per E1.20 SENSOR_DEFINITION) Recorded Value Support (1 byte per E1.20 SENSOR_DEFINITION) Description (0-32 bytes per E1.20 PARAMETER_DESCRIPTION)

(please be aware that this table omits any characters added by the SLIP encoding)

The UCM Module will send UCM RDM Sensor Define to as many of the generic UCM RDM Sensors as it needs for its own use (for example: Signal Level and Signal Quality). This then enables the Fixture to populate its local display with the values needed.

15.10. Synchronize Module-Specific Sensors

The UCM Module is now aware of the Sensor Number for each of its feedback values and can keep the Fixture updated via UCM RDM Sensor Set packages.

15.11. Additional Initialization Steps

The UCM Module can also obtain other information from the Fixture, if needed for its operation. For example: DMX Footprint, DMX Address & DMX Universe via GET DEVICE_INFO and GET ENDPOINT_TO_UNIVERSE.

16. Continuous Operation Guide

Once the initialization has been completed, the UCM Module will routinely perform the following operations:

- Send DMX frames to the fixture when available
- Pass on RDM frames from upstream controller towards Fixture (if relevant)
- Pass on RDM frames from Fixture towards upstream controller (if relevant)
- Perform RDM Get requests for module-specific parameters
- Update read-only module-specific parameters via UCM RDM PID Set
- Update module-specific sensors via UCM RDM Sensor Set

The following paragraphs will provide examples of each of these operations.

16.1. Send DMX Frames to the fixture when available

Whenever a UCM Module has a DMX Frame available, it will pass this on to the Fixture. A UCM Module may only populate the DMX Slots used by the Fixture at its current DMX Address with its current selected DMX Footprint and leave all remaining DMX Slots at 0x00.

Module to Fixture (DMX FRAME)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	0x0201 (513 bytes)
2	UCM Command Data	Universe Number
513	UCM Payload	Start Code 0x00 + 512 bytes of DMX slot data

(please be aware that this table omits any characters added by the SLIP encoding)

16.2. Pass on RDM frames

Whenever an upstream RDM controller (connected to the UCM Module) sends a message to the host fixture, the UCM Module will receive it and relay it over the USB connection. If the message is not broadcast, it expects an RDM to be sent back over the USB connection.

16.3. Perform RDM Get requests for module-specific parameters

The UCM Module will monitor the QUEUED_MESSAGE counter and issue corresponding RDM Get requests (which are also forwarded to the upstream RDM controller). If any of the modified parameters is a Module-Specific parameter, the UCM Module will also update this internally and adjust its behavior (for example: change radio channel, change pin-code, ...).

Module to Fixture (RDM: GET MODULE-SPECIFIC PARAMETER)

Bytes	Field	Value
1	UCM Start Code	0x00

1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	0x001A (26 bytes)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
26	UCM Payload	RDM: GET MODULE-SPECIFIC PARAMETER

(please be aware that this table omits any characters added by the SLIP encoding)

Fixture to Module (RDM: MODULE-SPECIFIC PARAMETER RESPONSE)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x01 (DMX/RDM Frame)
2	UCM Length	Variable (depending on parameter)
2	UCM Command Data	0x0000 (Universe 0 = don't care for RDM)
Length	UCM Payload	RDM: MODULE-SPECIFIC PARAMETER RESPONSE

(please be aware that this table omits any characters added by the SLIP encoding)

16.4. Update read-only module-specific parameters via UCM RDM PID Set

If the UCM Module has any read-only RDM Parameters (for example: connection state), it will update the Fixture when its state changes.

Module to Fixture (UCM RDM PID Set)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x02 (RDM PID Set)
2	UCM Length	Variable (depending on parameter)
2	UCM Command Data	RDM PID
Length	UCM Payload	Depending on parameter

(please be aware that this table omits any characters added by the SLIP encoding)

16.5. Update module-specific sensors via UCM RDM Sensor Set

If the UCM Module has any RDM Sensors (for example: signal quality), it will update the Fixture every second via a UCM RDM Sensor Set.

Module to Fixture (UCM RDM Sensor Set)

Bytes	Field	Value
1	UCM Start Code	0x00
1	UCM Command	0x03 (RDM Sensor Set)
2	UCM Length	Variable (depending on sensor)
2	UCM Command Data	RDM Sensor Number
Length	UCM Payload	Depending on sensor

(please be aware that this table omits any characters added by the SLIP encoding)