Extending RDM through the use of manufacturer-specific parameters

BY SIMON NEWTON

The website http://rdm.openlighting.org contains an index of known manufacturer-specific parameters to benefit the advancement of RDM within the industry.

ANSI E1.20, THE REMOTE DEVICE MANAGEMENT (RDM) PROTOCOL, provides a mechanism for bi-directional data exchange between lighting consoles and fixtures, on the same cable and using the same two conductors (pins 2 and 3) as DMX512. It offers extended control of lighting devices, far beyond what plain-vanilla DMX512 supports. The standard has some limitations in its current form, so a recently formed community project aims to address at least some of those limitations. The entertainment lighting community can participate in this project, and add to the power of RDM, without breaking the standard or working through the lengthy process of revising it.

A review of RDM

RDM enables a controller (a console) to discover responders (luminaires, et cetera) attached to a DMX512 line. Once a controller has discovered a responder, it can query and modify properties of the responder. Examples of such properties include the DMX512 start address, the fixture's personality, current operating temperature, and the fan speed. Some properties are read only; others can be modified. RDM uses GET commands to fetch the value of a property and SET commands to modify the value of a property.

The RDM protocol transports data as structured messages called *parameters*. Since multiple properties can be contained within a single message, parameter data may be composed of multiple data fields. A *parameter definition* describes the structure of the message, including the number of fields and the type and size of each field. Parameters have a unique 16-bit identifier, called a Parameter ID or PID, which is within the range 0 to 0xFFFF.

An example of a parameter is DMX_PERSONALITY, which has a PID of 0x00E0. When requested with a GET command, the DMX_PERSONALITY parameter returns the current personality of the fixture as well as the total number of personalities for that fixture. A SET DMX_PERSONALITY command can be used to change the current personality of the fixture.

The diagram in **Figure 1** illustrates an exchange of information between a controller and a responder. The controller sends a GET DMX_PERSONALITY request to the responder and the responder replies with a GET DMX_PERSONALITY response, containing two bytes of data.

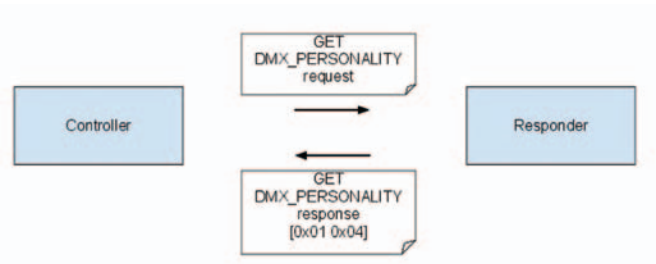

Figure 1 – An example of an RDM request and response exchange.

Figure 2 shows the parameter definition for the GET DMX_PERSONALITY response. The response data consists of two fields, each one byte in size. The value of the first byte is the current DMX512 personality and the value of the second byte the number of personalities available for the fixture.

Current Personality unsigned byte	Number of Personalities unsigned byte

Figure 2 – Parameter definition for a GET DMX_PERSONALITY response

With the parameter definition, the data returned by the responder now makes sense. In this example the responder supports four personalities, and the first personality is active.

Parameters allow the controller and the responder to agree on the format of data contained within the message. Without them, the data would simply be a collection of bytes with an undefined meaning. Unlike RDM, DMX512 does not specify the structure of its data, leaving the slot-to-property mapping to other means, usually a fixture library or an unfortunate person.

The RDM standard provides a base set of parameters, called the ESTA PIDs, for common functionality, such as the device

name, manufacturer's name, and lamp hours. The standard also supports the use of manufacturer-specific parameters, allowing manufacturers to extend RDM by adding functionality specific to their product. Manufacturer-specific parameters enable manufacturers to differentiate their product from others on the market. The manufacturer-specific parameter IDs are assigned within the range 0x8000 to 0xFFDF.

A problem arises when a controller from one manufacturer is to be used with responders made by a different manufacturer.

Limitations of manufacturer-specific PIDS

A problem arises when a controller from one manufacturer is to be used with responders made by a different manufacturer. The controller and responder will both agree on the format for the ESTA parameters, but the controller will need a way to obtain the parameter definition for any manufacturer-specific parameters that are to be used.

Describing the format of these manufacturer-specific PIDs is problematic. The RDM standard provides a PARAMETER_DESCRIPTION parameter, which was intended to be used for this purpose, but this method suffers from a number of deficiencies.

The PARAMETER_DESCRIPTION PID contains the following fields:

- Parameter message size—the total size of the message data.
- Data type—the type of data within the message. The type specifies the size of each data field, which allows the number of fields to be inferred.
- Unit—the SI unit for the data, e.g. Celsius
- Prefix—a multiplier to be applied to the value of this parameter e.g. kilo (x 10^3)
- Minimum Valid Value—the smallest value that can be used with this parameter
- Maximum Valid Value—the largest value that can be used with this parameter
- Default Value—the default value of the parameter
- Description—a text description of the parameter

This information is sufficient for describing simple message structures that contain a single data type, or a fixed-size list of homogeneous data types. However, messages that consist of different data types or variable-sized lists cannot be described. Ranges cannot be described on a per field basis, which means any range restrictions are assumed to apply to all fields.

Enumerated values are also not supported. This becomes an issue as a value of 1 for a hypothetical DEVICE_MODE parameter is meaningless without the added information that value 1 means "DMX512."

The PARAMETER_DESCRIPTION format also suffers from the assumption that GET responses and SET requests will always have the same format. It also does not provide a way of indicating that a GET request or SET response contains data.

The PARAMETER_DESCRIPTION PID is too limited to describe adequately anything beyond basic, single-property, parameters. Users often have to consult the product literature to determine the correct byte sequence to enter into an input field. The extent of the problem is increasing, as more manufacturers adopt manufacturer-specific parameters to add functionality to their products.

. . . manufacturer-specific parameters provide an important way to add value to a product and need to be supported as a first-class citizen in RDM.

One possible solution to overcome these limitations is to define a new parameter that provides a richer mechanism to describe RDM parameters. Such a standard would take years to complete, and there has not been much enthusiasm thus far from members of the RDM community to commence such an effort. Many are worried that an attempt to describe manufacturer-specific PIDs will produce a result as complicated as ACN's Device Description Language (DDL), which ANSI E1.17-2010 describes in an 88-page document.

A solution

I believe that manufacturer-specific parameters provide an important way to add value to a product and need to be supported as a first-class citizen in RDM.

In an effort to add better support of manufacturer-specific parameters in my own software, I started documenting the format of manufacturer-specific parameters at the beginning of 2011 and have made this data publicly available. The website http://rdm.openlighting.org contains an index of 170 known manufacturer-specific parameters along with the data format of each.

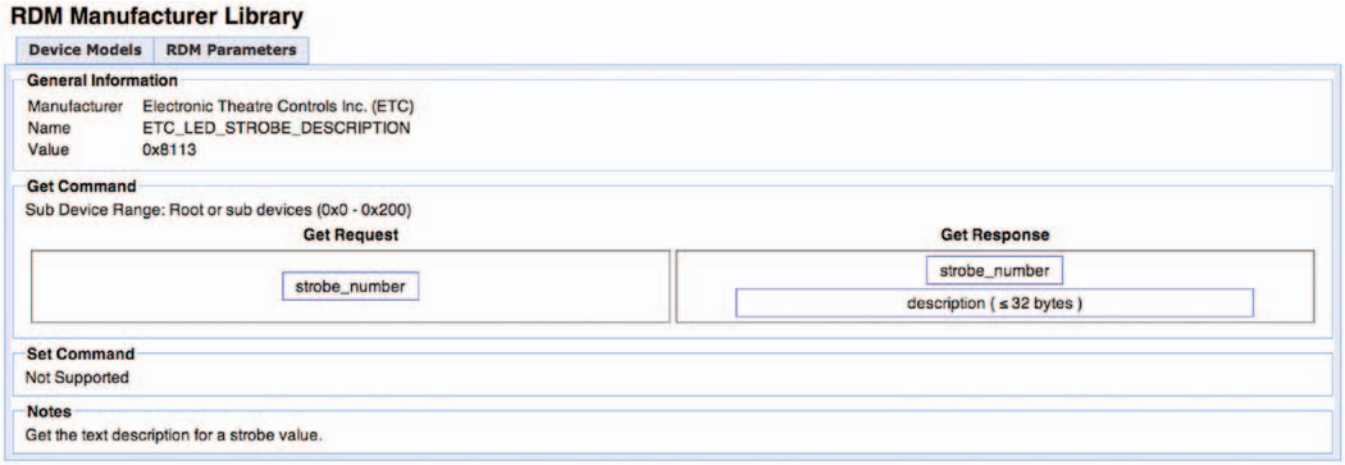

Figure 3 – The Parameter Definition for the ETC_LED_STROBE_DESCRIPTION PID

Figure 3 shows the definition of a manufacturer-specific parameter which controls the strobe settings for an LED fixture. Note that as this parameter contains data in the GET request message, it cannot be adequately described using PARAMETER_DESCRIPTION.

RDM controller manufacturers can now build easy-to-use interfaces for accessing manufacturer-specific PIDs, regardless of which manufacturer defined them.

The data format I have created addresses the limitations of the PARAMETER_DESCRIPTION PID and supports enumerated values, per field ranges, and repeated groups of non-homogeneous data types.

It is important to note that this effort does not attempt to create a single data format for describing manufacturer-specific parameters. The code behind the website is publicly available, and controller manufacturers are free to author plug-ins to export the parameter definitions in a format they find useful. This enables controller software to access the data, without being locked into a specific data format. RDM controller manufacturers can now build easy-to-use interfaces for accessing manufacturer-specific PIDs, regardless of which manufacturer defined them. I suspect most controller applications will automatically check online for parameter updates, freeing the user from ever having to deal with unknown manufacturer parameters again.

rdm.openlighting.org—More than just a parameter index

The rdm.openlighting.org site contains more than just parameter data. It also includes a list of RDM responders and, where available, the supported parameters, personalities, and sensor information of each. Images and links to the manufacturer's website for each product are also maintained.

The site provides end-users with an easy way to gauge RDM support in the industry and to research products they may be interested in. As of December 2011, the index contains over 170 device models, evidence of the increasing adoption of RDM within the industry.

The model data can be gathered from a Python script distributed with the Open Lighting Architecture (OLA). Once obtained, the data can be inserted into the data store by one of the site administrators.

The process will be streamlined further so that manufacturers will be able to collect the information themselves and, once happy with the data, publish it to the index. The product data is also fed to the new www.rdmprotocol.org website, which provides an alternative interface.

Please contact Simon Newton at simon@nomis52.net if you would like product data to be added.

So far, the parameter definitions are exported as Python data structures, but several manufacturers have expressed interest in an XML format. By iterating on a format outside of the standards body, the RDM community can make rapid progress, without committing ourselves to any one solution. My hope is that once the industry gains experience describing manufacturer-specific PIDs, a standards effort will proceed smoothly.

Some manufacturer-specific parameters such as SET_UID and ENABLE_DEBUG should not be exposed to the end-user, so keeping these parameter definitions private is desirable. Manufacturers can decide themselves which of their parameters are published to the index.

An important by-product of this effort is that it enables the community to recognize parameters from different manufacturers that perform the same function. These parameters are good candidates for subsequent standardization efforts. An example of such PIDs are IPv4 configuration parameters, and members of the RDM task group have started discussing the development of a standard set of parameters for this purpose.

Manufacturer-specific parameters are an important part of the RDM standard, as they allow manufacturers to differentiate their products. Unfortunately, the PARAMETER_DESCRIPTION PID is unable to describe completely many of the manufacturer-specific parameters in use today, so I have created an alternative. I think it's a useful resource now, but with the support of others in the entertainment lighting community, it can become even better. ■

Simon Newton has been interested in lighting control systems since middle school and founded the Open Lighting Project in 2004 with the aim of accelerating the adoption of new control protocols by the industry. He is a member of the Control Protocols Working Group and contributes to the RDM and RDM over IP standards. His day job finds him designing and building the serving infrastructure for a large Internet company in Silicon Valley. Simon can be reached at nomis52@gmail.com.