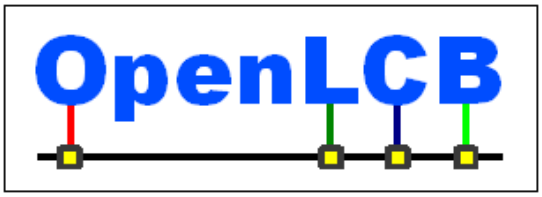

OpenLCB Technical Note	
Unique Identifiers	
Feb 15, 2015	Adopted

1 Introduction

This technical note contains informative discussion and background for the corresponding "OpenLCB Unique Identifiers Standard". This explanation is not normative in any way.

2 Annotations to the Standard

This section provides background information on corresponding sections of the Standard document. It's expected that two documents will be read together.

2.1 Introduction

Originally, OpenLCB was defined in terms of 48-bit unique "Node IDs". Other uses were found for these IDs that didn't associate them with a specific node, which demonstrated the need for Unique Identifiers beyond just Node IDs. Therefore, the Standard and this Technical Note are generally written in terms of Unique Identifiers, which includes Node IDs usage.

2.2 Intended Use

The "globally unique" requirement only refers to the universe of connected nodes; nodes that never need to communicate with each other need not have separate Node IDs. In general, however, nodes can move: they can be sold or loaned for use on another layout, nodes on modular layouts can be connected to other arbitrary modules, and few assumptions can be made. It is best if nodes are given a completely unique identifier when manufactured, so there's no need to ever detect and resolve a conflict. If a Node ID conflict should occur on an OpenLCB network, the Message Network Standard defines how the conflict should be handled. In general, OpenLCB standards place the effort and burden on preventing a conflict from occurring rather than resolving a conflict if and when it does occur.

2.3 References and Context

2.4 Format

The Standards don't require any particular human-readable format for input and output, but hex-pairs with separators (example: 01.AB.34.01.CD.E3) are recommended by the Common Information TN. If any other format is used, including decimal pairs, it's very important to make it clear how to interpret it.

There are many methods to store a Unique Identifier, and that's not constrained by this Standard. It could be stored in a non-volatile memory, as jumpers on a board, etc.

2.5 Allocation

2.5.1 Overview

Unique Identifiers are assigned via a delegation process. At the highest level, ranges are assigned to people and organizations, within which they are responsible for assigning unique identifiers to separate devices. These ranges can be subdivided and delegated further, as needed. Additional ranges can then be requested, which will eventually be recorded in the Unique Identifier Standard. The OpenLCB organization reserves the right to allocate Unique Identifiers within a separate database in real-time and periodically update the Unique Identifier Standard document from this database. In the unlikely event that a conflict shall arise between OpenLCB managed real-time database and the Unique Identifiers Standard, the Unique Identifiers Standard, the Unique Identifiers Standard document represents the true allocation.

One of the reasons for having a long, 48-bit Unique Identifier space is to make it easier to use a delegation system like this. Because there are a lot of possible Unique Identifiers, large ranges can be delegated to groups without having to ensure that the range be efficiently used. For example, most NMRA members will not design their own OpenLCB nodes and need to assign node IDs, but assigning a range to every member makes it easy for those who want to, at a total cost of less than 0.0016% of the available Unique Identifier space.

In the delegated assignments, the lower order byte(s) are self-assigned. The value of zero should be reserved, indicating that a number within the range hasn't been assigned.

The high byte of each range is different for each type of assignment, making it easy to determine the allocation pattern in use for a particular Unique Identifier.

Allocations are meant to be unique forever, so the standard requires that new allocation ranges not overlap existing ones, and allocation regions not be reused later.

2.5.2 Reserved Leading Zero

A Unique Identifier with a most significant value of 0x00 is never valid. This range is reserved. A message utilizing one of these values must never be sent on the OpenLCB bus. If a node receives a message utilizing a Unique Identifier from this range, it may ignore the message, or throw an error, however, it must never act on the message as if it was a valid Unique Identifier.

The fact that the Unique Identifiers in this range are invalid may be, and often is, exploited within the internal software of a node in order to mark an unused, or invalid, Node ID. This use is private and internal to the node and as such is not subject to scrutiny by the OpenLCB standards as long as it is not exposed external to the node.

2.5.3 Well-Known Global Identifiers

In addition to use as node identifiers (Node IDs), OpenLCB unique identifiers are used to ensure uniqueness of specific global event identifiers and for other purposes. These numbers must be allocated so that they are kept unique. The identifiers specified in this section are of that type.

Note that the detailed use of these identifiers is specified elsewhere. In some cases, the protocols are still being developed, and the entry here is just reserving a range for a specific future use.

2.5.4 Manufacturer Specific

This group of Unique Identifiers is reserved specifically for manufacturers. Manufacturers may request a range of Unique Identifiers for use within their products. A Manufacturer may assign the Unique Identifiers within their allocated range at their own discretion so long as every assignment is unique.

A manufacturer is not required to assign Unique Identifiers sequentially. They may choose their own arbitrary scheme that could be based on product type, manufacture date, or some other method of their choosing. Though it is highly desirable for a manufacturer to make reasonably efficient use of their self-assigned space, should a manufacturer require additional space, it may be requested, and granted, without the requirement of having used every last available Unique Identifier within a previously assigned space. Manufacturers are not required to publicly disclose their allocation scheme for Unique Identifiers.

In order to encourage existing manufacturers to participate in developing OpenLCB products, every DCC manufacturer has already been allocated a 24-bit region within which to self-assign Unique Identifiers from. The DIY, JMRI, MERG, and NMRA Reserved spaces are sub-sets of the DCC manufacturer space and are consistent with the DCC ID's already assigned for these purposes in the NMRA standards doc “S-9.2.2 Appendix A, DCC Manufacturer ID Codes”. They are singled out only for the purpose of drawing attention to the fact that they exist and not for any other purpose.

2.5.5 Self-Assigning Groups

MERG kit builders and others would like to assign their own identifiers without going through a complicated process. To make this possible without any interaction with anybody, these groups are assigned identifier ranges that involve their member number in the organization. A member may assign any identifier from this range to the node(s) they produce, provided that each identifier is assigned to at most one node. A range of 255 identifiers per member is typically sufficient for hobby usage. Should a hobbyist exhaust their assigned range, the hobbyist can get another, larger assignment. It's also convenient to give hobbyists a byte as their range.

Each organization is assigned a unique high-order two bytes. The organization member number is given 24 bits. Byte 2 of the Unique Identifier advances by 4 between groups (NMRA is 0x00, MERG is 0x04, etc) to allow a little more headroom on group membership numbers; this space can be reclaimed later if needed.

Other groups have defined mechanisms to ensure that their node numbers or equivalent constructs are uniquely assigned. They may have non-technical reasons for wishing to use those same mechanisms to assign OpenLCB unique identifiers. Ranges of OpenLCB Unique Identifier ranges can be assigned to these groups, so that when people then use the group's mechanism to select a value within that range, the result will be a properly unique OpenLCB Unique Identifier.

The first example of this is MERG CBUS developers. MERG CBUS has defined a “no cost” way of identifying unique 16-bit Node Number (NN) for CBUS use, perhaps with an optional 16-bit Layout Number (LN). People who wish to use this mechanism to allocate unique OpenLCB Node ID identifiers can, without having to consult anybody, generate an OpenLCB Node ID from the unique CBUS number(s) as described in the Standard.

If the user is involved in determining the Unique Identifier for a node (the Node ID) by setting switches, the possibility of duplicated Node IDs must be considered. Users make mistakes. To reduce

user frustration, the node should provide a user-visible way to indicating a duplicate has been seen, and should fully implement the relevant wire-protocol-specific methods for detecting duplicate Node IDs.

2.5.6 Assigned by Software at Runtime

Programs that act as one or more OpenLCB nodes need to associate unique identifiers with them. For licensed software, where a unique key can be associated with each instance of the program, this is easy: Use the manufacturer space defined above, and generate the lower bits of a specific ID from the license key.

Free, open and unlicensed software can't use a license-key-based method. Unfortunately, the 48-bit address space is too small to use the IP-address-plus-signature GIDs that would otherwise make this a simple problem, or the even larger MAC-address-plus-signature GIDs.

Initial experiments were done using 32-bit IPv4 addresses as components of Unique Identifiers, but this is no longer recommended for several reasons:

- Not all IPv4 addresses are globally unique. Some IP addresses correspond to “private networks”, which are only locally unique. See RFC 1918 and RFC 3330 for more information. In addition, Microsoft defined a non-IETF “Automatic Private IP Addressing” mechanism for providing non-global IP addresses.
- A single computer may run several programs, so there still needs to be separate mechanism to provide a unique value for the lowest bytes of the ID. That involves a level of coordination across multiple software vendors that's hard to imagine.
- IPv6 is coming. It provides addresses that are too large to use directly. Even before that happens, the various issues of IPv4 to IPv6 mapping raise all sorts of questions about uniqueness of IPv4 addresses.
- Even globally routable IPv4 addresses may not be unique over time. For example, DHCP may assign the same address to multiple computers sequentially. This is particularly an issue with wireless access at e.g. clubs and shows.

Computers that have global Internet access, even if they don't have a permanent and unique IP address, can still get a Unique Identifier from an openlcb.org-provided service. These Unique Identifiers are provided from a specified range to ensure that they are unique when created. Each identifier is only provided once to ensure that it remains unique. Programs using this facility should permanently remember Unique Identifiers obtained this way, because they won't get the same one on a later request.

Other organizations can also distribute Unique Identifiers from within their allocated blocks. For example, a Unique Identifier could be provided when a free-software program is downloaded, perhaps as part of the download package or even as part of its filename.

Programs without access to an ID-providing service must use some other mechanism, which may result in prompting the user for a Unique Identifier assigned by one of the other mechanisms.

2.5.7 Specifically Assigned by Request

Users can request blocks of Unique Identifiers of various sizes. The small (256) and medium (65536) blocks are not scarce resources. Requests for these blocks should be routinely granted once the

requester has been identified. The 24-bit blocks are slightly scarcer, but there are still almost 2^{16} of them available by using additional values for byte 2.

An initial automated system for requesting and obtaining unique ID ranges is available at http://openlcb.org/trunk/web/requestuidrange.php and subsequent pages.

2.5.8 Locomotive Control Systems

Locomotive control was initially beyond the scope of OpenLCB development, but later work started to define OpenLCB methods for working with existing locomotive control systems. This section specifies ranges of Unique Identifiers that are reserved for the purpose of interfacing with locomotive control. The details of how they are to be used are specified elsewhere, but sufficient range has been reserved to allow providing locomotive control systems with Unique Identifiers. It is important to note that at the time of this document's publication, locomotive control does not require any of these Unique Identifiers. For this reason, this Unique Identifier range is considered deprecated. Do not use these Unique Identifiers for any reason, they are no longer valid.

2.5.9 RFID and NFC

The RFID and NFC Unique Identifiers space is reserved to be used in future standards to be defined elsewhere.

2.5.10 Temporary Assigned by Software at Runtime

As described in the section 2.5.6 Assigned by Software at Runtime above, there are many reasons for which a node may want to be assigned a Unique Identifier at runtime. This Unique Identifier space is specifically designated for temporary (leased) assignment of these Unique Identifiers. There is a contract implied with the assignment between the assigning server and client node that is valid for a time period defined by the server. Once the prescribed time period expires, the client node must cease usage of the previously assigned Unique Identifier.

It is left up to the user to guarantee that there are not two or more servers with the potential to assign the same Unique Identifier within a single OpenLCB network.

The implementation of the client server protocol is not explicitly defined, and currently there is no OpenLCB protocol that allows this client/server process to run over an OpenLCB network. Other mechanisms, such as a TCP connection over the Internet may be used. The following list suggests implementation details that are not required, but might be considered when designing such a client server pair:

- Lease time can be extended through a renewal process.
- The server does not reassign a previously assigned Unique Identifier until it has run through the entire pool allocated to it.
- The server allows the client to suggest a Unique Identifier that it would like from the server's pool.
- The server have persistent storage of current leases that it can can consult in case of a restart.
- The client remembers its last Unique Identifier assignment and suggests to the server to be re-assigned this Unique Identifier upon reset, power failure, or reconnection to the server following a previous disconnect.

Use this pool with caution. Implementation mechanisms that make use of this pool are currently experimental.

2.5.11 Reserved Unique Identifiers

For error detection, we permanently reserve all identifiers that start with either a 0x00 or 0xFF value. OpenLCB implementations should, but are not required to, treat it as an error when any of those are encountered.

3 Implementation Information

Specific Unique Identifier assignments are stored in a MySQL database on the http://openlcb.org website. The full list of assignments, include overall ranges from the standard and ranges assigned for specific purposes and users, can be found on the http://www.openlcb.org/trunk/web/viewuid.php dynamic page.

Automated allocation systems can be abused, and we don't want to give away large chunks of address space to automated requesters. All available information about requests is logged. Users are asked for their name and contact information at the time of the request, which is also logged. Depending on our experience with requests, we may have to add an email challenge-response or other mechanism to ensure only valid requests get allocations.

Table of Contents