OpenLCB Technical Note	
OpenLCB Event Identifiers	
Jan 24, 2013	Preliminary

1 Introduction

This explanatory note contains informative discussion and background for the corresponding "OpenLCB Event ID Specification". 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

2.2 References and Context

2.3 Format

The specification doesn't require any particular human-readable format, but hex-pairs with separators are strongly suggested, e.g. 01.AB.34.01.CD.E3.20.0E; decimal pairs could also be used, but in that case it's important to provide a way to know which is use.

3 (Text to be integrated)

Below this is just a collection of pieces from other docs right now.

The proposed OpenLCB depends on globally unique Event ID numbers (EIDs). This node discusses the assignment of those.

The "globally unique" requirement only refers to the universe of connected nodes; nodes that never need to communicate with each other don't need to have separate Event 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. Therefore, we require globally uniqueness for all EventIDs.

To ensure uniqueness, the top 6 bytes of EventIDs that you define are required to be within the NodeID space that you control. The low two bytes would be any number you prefer, so long as you use each value for only one event.

This requirement applies equally to events defined by a hardware node, e.g. by button pushing, a software node in a computer, or EventIDs that are defined by a human writing them on a piece of

paper. In each case, the thing doing the definition must ensure it has control over the NodeID corresponding to the top four bytes, so it can ensure that the EventID not be reused.

Table 1: Event Number Construction

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
Node ID assigned to you						Unique 16 bit extension	

The Node ID part could be e.g. from real nodes that you own. For example, the Blue/Gold algorithm for assigning Event IDs automatically puts the Node ID of the first node to use that event in the Event ID, guaranteeing it to be unique.

Not all Event IDs need to be assigned through that algorithm. For example, a software configuration tool might define events and assign Event IDs to them. A board manufacturer may prefer some other pushbutton configuration process. A modular club may decide that certain events form the “boundaries” of modules, and need to be assigned well-known Event IDs to make it easier to create large modular layouts. Whatever the method, the Event IDs need to be globally unique, and we ensure that by requiring that people create EventIDs using node ID numbers assigned to them (and therefore not assigned to anybody else), plus an additional 16 bits that they are responsible for using only once.

Note that the Node ID part of an Event ID doesn't have to correspond to any physical node. So long as you've got some Node ID address space assigned to you, you can use that. For example, a fast-clock manufacturer might want to define a range of EventIDs so that a different event is emitted every fast minute. (This lets e.g. lamp controllers do day-night cycles automatically). For the sake of argument, let's say 24 bits worth of specific events are needed (it's actually smaller, but let's use this). The manufacturer has a large range of node IDs assigned already, so can use 8bits worth of that space, plus the extra 16 bits in the EventID below the nodeID-like-part, to do this. For example, if a manufacturer had a range of Node IDs that included at least 0x123456789A00 through 0x123456789AFF (in other words, all possible values of the low byte), he could create a set of Event IDs with the low 24 bits used to carry a time value:

Table 2: Example of defining a Set of Events

Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
Assigned Node ID						Unique 16 bit extension	
0x12	0x34	0x56	0x78	0x9A	Time Value		

This is guaranteed to be unique, no matter where one of these devices is taken, because the node ID range is guaranteed to belong to the manufacturer, and he'll use it only once.

3.1.1 Well-known Event ID extensions

Because events are a very convenient way to broadcast information to the entire network, we define some extension values to represent common conditions:

0xFFF8 – Node has logged a new log entry (See logging protocol note)

0xFFF7 -

(But is this mechanism needed? Global events would convey the same information, without using up event space, and the source node ID could be used to determine who said it.

"I've logged an entry", said Joe Bloggs

"Joe Bloggs has logged an entry", said Joe Bloggs.

Table of Contents