	OpenLCB Technical Note
	Method for Allocated MTI Values
Jul 23, 2011	**Preliminary**

1 Introduction

OpenLCB messages carry a Message Type Indicator (MTI) field to indicate their type. The primary form is a 16-bit number.

This note describes how the numeric values for those MTIs are allocated. The current allocations are documented in a separate spreadsheet[1]. Those allocations are normative. The discussion in this note is not normative on OpenLCB users or node developers, but does describe the methods that are to be used for allocating new MTI values for new OpenLCB message and protocol types.

MTI information may be carried in different formats for different wire protocols. These are documented in the same spreadsheet, and discussed in separate sections below.

2 Common MTI Values

Many nodes will treat MTIs as magic 16-bit numbers, just comparing them for equality to specific values of interest. That's a perfectly fine implementation strategy.

To allocate them, however, having a systematic pattern to the values can be more useful.

- The two most-significant bits are reserved as 00; nodes must send and check that value.

- The next bit is used to indicate unaddressed messages meant for "simple" or minimal nodes. A 1 in this bit means that these simple nodes can ignore this message. A 0 in this bit means that the simple nodes should process the message.
  This is meant to speed processing of some message types. The decision as to whether a particular message type is of interest to the notional "simple" node is left to the actual protocol definition, found elsewhere. This bit is reserved to 1 for addressed messages, as a message specifically delivered to a node should be processed.

- The next bit, the least significant of the top nibble, is reserved as 1; nodes must send and check that value.

- The next eight bits are used to number a specific message number. This also has some substructure:

  - The most significant bit is reserved as 0; nodes must send and check that value.

  - The next two of these are used to form static priority groups. A 0 bit is considered to have more priority (can be processed first), a 1 bit less priority (can be

---

[1]See http://openlcb.org/specs/index.html for the current version of the spreadsheet. It provides concrete examples that may help you understand the material in this document.

30 processed later). The MSB makes a larger statement about priority than the LSB of these two. Priority processing is permitted but not required. The priority group bits are part of the overall message type.

- The next bit is reserved as 0; nodes must send and check that value.

- The next 4 bits, forming the low nibble, are used to indicate the specific message type within a priority group.

35 - The bottom nibble is interpreted as flags that define the structure and format of the message type.

  - The most significant bit is reserved as 0; nodes much send and check that value.

  - The 2nd from-least-significant bit indicates that this message carries a destination node address (DID) when set to 1. Setting 0 means that the message is globally addressed. If a 40 Destination Node ID (DID) is present, it is at the specific location in the message content.

  - The next-to-least-significant bit indicates this message carries a P/C Event ID field when set to 1. Setting 0 means that the message does not carry a P/C event ID. If a P/C Event ID is present, it's at the start of the message, it is at the specific location in the message 45 content.

  - The least-significant bit when set to 1 indicates this message carries a flag byte after the DID and/or EID determined by the above bits. The low bits of that byte can be relocated in CAN messages, see the definition of the CAN wire protocol.

We've chosen to allocate bit fields to make decoding simpler; if possible, aligned on nibble boundaries 50 to make it easy to read as hexadecimal numbers.

MTI layout:

Field	Reserved	Simple Node flag	Reserved	Message Number	Reserved	Destination ID flag	Event ID flag	Flag Byte flag
Size & location	2 bits 0xC000	1 bit 0x2000	1 bit 0x1000	8 bits 0x0FF0	1 bit 0x0008	1 bit 0x0004	1 bit 0x0002	1 bit 0x0001
Value(s)	0 (send and check)	0 means for simple node, 1 means simple node can skip	1 (send and check)	See below	0 (send and check)	1 means Destination ID present, 0 means not present	1 means Event ID present, 0 means not present	0 means flag byte present, 1 means not present

Message Type byte (8 bits) layout:

Field	Reserved	Static priority groups	Reserved (send and check)	Specific type
Size & location (within 8-bit field)	1 bit 0x80	2 bits 0x60	1 bit 0x10	4 bits 0x0F
Value(s)	0 (send and check)	0 to 3 0 goes first, 3 last if priority processing is present	0 (send and check)	See spreadsheet for values & meanings

55   Specific values are allocated and documented in a separate worksheet. We keep them in just that one place to avoid conflicting updates.

3 CAN MTI Considerations

The standard CAN MTI field is 12 bits in the header (messages without destination address) or one byte in the data segment (addressed messages). Since CAN frames only carry 8 data bytes, a 1-byte

60   MTI short form will be used at first. The possibility of longer MTI values has been reserved. MTI information is carried in a different format on CAN links to increase bandwidth efficiency, simplify decoding in small processors, and permit use of hardware filtering.

Standard MTIs are mapped to one of seven types:

Type	Meaning
0	Unaddressed MTI for simple nodes
1	Unaddressed MTI for non-simple nodes
2	(Reserved, not used)
3	(Reserved, not used)
4	Datagram not-last segment
5	Datagram last segment

Type	Meaning
6	Addressed MTI for all nodes
7	Stream Data

65   The bit coding of this has been chosen to use the "4" bit to indicate that a destination address is present.

The 4, 5 (Datagram) and 7 (Stream Data) values are special cases chosen for efficient processing of large amounts of data on CAN. Most MTIs will map to 0, 1 or 6.

3.1 Addressed CAN MTIs

Common MTIs with the "destination ID present" bit set are mapped to and from type 6.

70   In this case, the destination alias is placed in the header, and the Message Type byte is the first byte of the CAN frame.

3.2 Unaddressed CAN MTIs

Common MTIs with the "destination ID present" bit set are mapped to and from type 0 or 1, depending on whether the "simple node" bit is reset or set.

75   In this case, the next twelve bits of the CAN header are available for MTI information. The first (most significant) eight bits are used for the Message Type byte. The last (least significant) are used for the Event ID present, and flag byte present flag, and any flag bits themselves.

Table of Contents