LAUNCHCONTROL

Programmer's Reference

1 Introduction

This manual describes Launch Control's MIDI communication format. This is all the proprietary information you need to be able to write patches and applications that are customised for Launch Control.

It is assumed that you already have a basic knowledge of MIDI, and some appropriate software for writing interactive MIDI applications (for example, Max for Live, Max/MSP, or Pure Data).

Numbers in this manual are given in both hexadecimal and decimal. To avoid any ambiguity, hexadecimal numbers are always followed by a lower-case h.

2 Launch Control MIDI Overview

Launch Control is a class-compliant USB device that boasts 8 pads and 4 buttons. Every pad contains a bi-coloured LED with a red element and a green element; the light from these elements can be mixed to form amber. Each button contains a single red LED. Launch Control has 16 templates: 8 user templates, which can be modified, and 8 factory templates, which cannot. User templates occupy slots 00h-07h (0-7), whereas factory templates occupy slots 08-0Fh (8-15). Use the Launch Control Editor (available on the Novation website) to modify your 8 user templates.

Launch Control has a single MIDI port named 'Launch Control *n*', where *n* is the device ID of your unit (not shown for device ID 1). The pad LEDs for any template can be controlled via System Exclusive messages. Alternatively, pad LEDs for the currently selected template can by controlled via MIDI note-on, note-off, and control change (CC) messages, as per the original Launchpad protocol.

Launch Control uses a System Exclusive protocol to update the state of any pad on any template, regardless of the currently selected template. In order to maintain compatibility with Launchpad and Launchpad S, Launch Control also adheres to the traditional Launchpad LED lighting protocol via note-on, note-off and CC messages. However, such messages will only be acted upon if the currently selected template contains a pad whose note/CC value and MIDI channel match those of the incoming message. Users are therefore advised to adopt the new System Exclusive protocol.

In addition, Launch Control also supports the original Launchpad double-buffering, flashing and set-/reset-all LED messages, where the MIDI channel of the message defines the template for which the message is intended. These messages can therefore be sent at any time, regardless of which template is currently selected.

The state of each LED is stored when the template is changed and will be recalled when the template is reselected.

3 Computer-to-Device Messages

LEDs on the Launch Control can be set via two different protocols: (1) the traditional Launchpad MIDI protocol, which requires the currently selected template to contain a pad whose note/CC and MIDI channel correspond to the incoming message; and (2) the Launch Control System Exclusive protocol, which will update the required pad regardless of its note/CC value or MIDI channel.

In both protocols, a single byte is used to set the intensities of both the red and green LEDs. This byte also includes the *Copy* and *Clear* flags. The byte is structured as follows (those unfamiliar with binary notation can read on for the formula):

Bit	Name	Meaning
6		Must be 0
5..4	*Green*	Green LED brightness
3	*Clear*	If 1: clear the other buffer's copy of this LED
2	*Copy*	If 1: write this LED data to both buffers Note: this behaviour overrides the *Clear* behaviour when both bits are set
1..0	*Red*	Red LED brightness

The *Copy* and *Clear* bits allow manipulation of the Launch Control's double-buffering feature. See the 'Control double-buffering' message and the Appendix for details about how this can be used.

Each LED can therefore be set to one of four values:

Brightness	Meaning
0	Off
1	Low brightness
2	Medium brightness
3	Full brightness

If the double-buffering features are not in use, it is good practice to keep the *Copy* and *Clear* bits set when turning LEDs on or off. This makes it possible to use the same routines in flashing mode without re-working them. A formula for calculating velocity values is:

Hex version *Velocity* = (10h x *Green*)
+ *Red*
+ *Flags*

Decimal version *Velocity* = (16 x *Green*)
+ *Red*
+ *Flags*

where *Flags* = 12 (OCh in hex) for normal use;
8 to make the LED flash, if configured;
0 if using double-buffering.

The following tables of pre-calculated velocity values for normal use may also be helpful:

Hex	Decimal	Colour	Brightness
0Ch	12	Off	Off
0Dh	13	Red	Low
0Fh	15	Red	Full
1Dh	29	Amber	Low
3Fh	63	Amber	Full
3Eh	62	Yellow	Full
1Ch	28	Green	Low
3Ch	60	Green	Full

Values for flashing LEDs are:

Hex	Decimal	Colour	Brightness
0Bh	11	Red	Full
3Bh	59	Amber	Full
3Ah	58	Yellow	Full
38h	56	Green	Full

Launchpad Protocol

Note On — Set pad/button LEDs

Hex version 9*n*h, *Note*, *Velocity*
Dec version 144+*n*, *Note*, *Velocity*

A note-on message changes the state of all pads/buttons in the currently selected template whose note/CC value matches that of the incoming *Note* value and whose zero-indexed MIDI channel matches the MIDI channel *n* of the incoming message. *Velocity* is used to set the LED colour.

Note Off — Turn off pad/button LEDs

Hex version 8*n*h, *Note*, *Velocity*
Dec version 128+*n*, *Note*, *Velocity*

This message is interpreted as a note-on message with the same *Note* value but with a velocity of 0. The *Velocity* byte is ignored in this message.

Reset Launch Control

Hex version Bnh, 00h, 00h
Dec version 176+*n*, 0, 0

All LEDs are turned off, and the buffer settings and duty cycle are reset to their default values. The MIDI channel *n* defines the template for which this message is intended (00h-07h (0-7) for the 8 user templates, and 08h-0Fh (8-15) for the 8 factory templates).

Control double-buffering

Hex version Bnh, 00h, 20-3Dh
Dec version 176+*n*, 0, 32-61

This message is used to control the double-buffering state of the pads. The MIDI channel *n* defines the template for which this message is intended (00h-07h (0-7) for the 8 user templates, and 08h-0Fh (8-15) for the 8 factory templates). See the Appendix for more information on double-buffering. The last byte is determined as follows:

Bit	Name	Meaning
6		Must be 0.
5		Must be 1.
4	*Copy*	If 1: copy the LED states from the new 'displayed' buffer to the new 'updating' buffer.
3	*Flash*	If 1: continually flip 'displayed' buffers to make selected LEDs flash.
2	*Update*	Set buffer 0 or buffer 1 as the new 'updating' buffer.
1		Must be 0.
0	*Display*	Set buffer 0 or buffer 1 as the new 'displaying' buffer.

For those less familiar with binary, the formula for calculating the data byte is:

Hex version *Data* = (4 x *Update*)
+ *Display*
+ 20h
+ *Flags*

Decimal version *Data* = (4 x *Update*)
+ *Display*
+ 32
+ *Flags*

where *Flags* = 16 (10h in Hex) for *Copy*;
8 for *Flash*;
0 otherwise.

The default state is zero: no flashing; the update buffer is 0; the displayed buffer is also 0. In this mode, any LED data written to Launch Control is displayed instantly.

Sending this message also resets the flash timer, so it can be used to resynchronise the flash rates of all Launch Controls connected to a system.

Turn on all LEDs

Hex version	B*n*h, 00h, 7D-7Fh
Dec version	176+*n*, 0, 125-127

The last byte can take one of three values:

Hex	Decimal	Meaning
7Dh	125	Low brightness test.
7Eh	126	Medium brightness test.
7Fh	127	Full brightness test.

Sending this command resets all other data — see the *Reset Launch Control* message for more information. The MIDI channel *n* defines the template for which this message is intended (00h-07h (0-7) for the 8 user templates, and 08h-0Fh (8-15) for the 8 factory templates).

Launch Control System Exclusive Protocol

Set pad/button LEDs

System Exclusive messages can be used to set the red and green LED values for any pad in any template, regardless of which template is currently selected. This is done using the following message:

Hex version	F0h 00h 20h 29h 02h 0Ah 78h *Template LED Value* F7h
Dec version	240 0 32 41 2 10 120 *Template LED Value* 247

Where *Template* is 00h-07h (0-7) for the 8 user templates, and 08h-0Fh (8-15) for the 8 factory templates; *LED* is the index of the pad/button (00h-07h (0-7) for pads, 08h-0Bh (8-11) for buttons); and *Value* is the velocity byte that defines the brightness values of both the red and green LEDs.

Toggle button states

The state of buttons whose behaviour is set to 'Toggle' (rather than 'Momentary') can be updated by System Exclusive messages. This is done using the following message:

Hex version	F0h 00h 20h 29h 02h 0Ah 7Bh *Template LED Value* F7h
Dec version	240 0 32 41 2 10 123 *Template LED Value* 247

Where *Template* is 00h-07h (0-7) for the 8 user templates, and 08h-0Fh (8-15) for the 8 factory templates; *LED* is the index of the pad/button (00h-07h (0-7) for pads, 08h-0Bh (8-11) for buttons); and *Value* is either 00h (0) for off or 7Fh (127) for on. Messages for buttons not set to 'Toggle' will be ignored.

Change current template

The following message can be used to change the current template of the device:

Hex version	F0h 00h 20h 29h 02h 0Ah 77h *Template* F7h
Dec version	240 0 32 41 2 10 119 *Template* 247

Where *Template* is 00h-07h (0-7) for the 8 user templates, and 08h-0Fh (8-15) for the 8 factory templates.

4 Device-to-Computer messages

Pad/button pressed

Hex version	9*n*h, *Note*, *Velocity*
Dec version	144+*n*, *Note*, *Velocity*

OR

Hex version	B*n*h, *CC*, *Velocity*
Dec version	176+*n*, *CC*, *Velocity*

Pads/buttons can output either note messages or CC messages on a zero-indexed MIDI channel *n*. A message is sent with velocity 7Fh when a button is pressed; a second message is sent with velocity 0 when it is released.

Template changed

Launch Control sends the following System Exclusive message out on changing template:

Hex version	F0h 00h 20h 29h 02h 0Ah 77h *Template* F7h
Dec version	240 0 32 41 2 10 119 *Template* 247

Where *Template* is 00h-07h (0-7) for the 8 user templates, and 08h-0Fh (8-15) for the 8 factory templates.

Appendix — LED double-buffering and flashing

The Launch Control has two LED buffers, 0 and 1. Either one can be displayed while either is updated by incoming LED instructions. In practice, this can enhance the performance of Launch Control in one of two ways:

1. By enabling a large-scale LED update which, although it could take 100 milliseconds to set up, appears to the user to be instantaneous.

2. By automatically flashing selected LEDs.

To exploit double-buffering for the first purpose requires very little modification to existing applications. It can be introduced in the following way:

1. Send B*n*h, 00h, 31h (176+*n*, 0, 49) on start-up, where *n* defines the template for which this message is intended (00h-07h (0-7) for the 8 user templates, and 08h-0Fh (8-15) for the 8 factory templates). This sets buffer 1 as the displayed buffer, and buffer 0 as the updating buffer. Launch Control will cease to show new LED data that is written to it.

2. Write LEDs to the Launch Control as usual, ensuring that the *Copy* and *Clear* bits are not set.

3. When this update is finished, send B*n*h, 00h, 34h (176+*n*, 0, 52). This sets buffer 0 as the displayed buffer, and buffer 1 as the updating buffer. The new LED data will instantly become visible. The current contents of buffer 0 will automatically be copied to buffer 1.

4. Write more LEDs to the Launch Control, with *Copy* and *Clear* bits set to zero.

5. When this update is finished, send B*n*h, 00h, 31h (176+*n*, 0, 49) again. This switches back to the first state. The new LED data will become visible, and the contents of buffer 1 will be copied back to buffer 0.

6. Continue from step 2.

7. Finally, to turn this mode off, send B*n*h, 00h, 30h (176+*n*, 0, 48).

Alternatively, chosen LEDs can be made to flash. To turn on automatic flashing, which lets Launch Control use its own flashing speed, send:

Hex version	B*n*h, 00h, 28h
Dec version	176+*n*, 0, 40

If an external timeline is required to make the LEDs flash at a determined rate, the following sequence is suggested:

Turn flashing LEDs on	B*n*h, 00h, 20h	(decimal version 176+*n*, 0, 32)
Turn flashing LEDs off	B*n*h, 00h, 20h	(decimal version 176+*n*, 0, 33)

As mentioned previously, it is good practice to keep the *Clear* and *Copy* bits set while addressing LEDs generally, so that an application can easily be expanded to include flashing. Otherwise, unintended effects will occur when trying to introduce it later.