LAUNCHPAD

Programmer's Reference

www.novationmusic.com

1 Introduction

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

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 equivalents, as different software favours the use of different conventions. To avoid any ambiguity, hexadecimal numbers are always followed by a lower-case h.

2 Launchpad MIDI Overview

Launchpad comprises eighty buttons. These divide into three sections: a square grid of sixty-four buttons, eight round 'scene launch' buttons arranged along the right-hand side, and a row of round buttons at the top that are generally employed by Automap or Live.

Every button is back-lit by a bi-coloured LED. Each LED consists of a red and a green element. When these are both turned on, the light can be mixed to form amber.

All communication with Launchpad is conducted using MIDI note-on, note-off, and controller change messages. Launchpad transmits and receives on MIDI channel 1. There is one exception to this, which will be covered later, but it is not essential to learn it.

Hence a Launchpad MIDI message is always three bytes long. (For good reasons, the driver does not support running status.) A valid message therefore takes one of these forms:

Message type	**Hex version**	**Decimal version**
Note off	80h, *Key*, *Velocity*	128, *Key*, *Velocity*
Note on	90h, *Key*, *Velocity*	144, *Key*, *Velocity*
Controller change	B0h, *Controller*, *Data*	176, *Controller*, *Data*

Launchpad uses a low-speed version of USB. A limitation of this is that it accepts a maximum of 400 messages per second. Because there are 80 LED addresses (one for each bi-colour LED), it will take 200 milliseconds to update a Launchpad completely. Two work-arounds are provided to speed up its real and apparent update speed:

- MIDI channel 3 note-on messages (beginning 92h, or 146 decimal) can be used to update the entire surface two LEDs at a time.
- Launchpad can be double-buffered. This means that all the LED states can be updated internally while they continue to show their existing state. The buffers may then be swapped with a single command, so that the update of the surface appears to be instantaneous.

3 Computer-to-Launchpad Messages

Note Off

Hex version	80h, *Key*, *Velocity*
Decimal version	128, *Key*, *Velocity*

This message is interpreted in exactly the same way as a note-on message containing the same key code, and velocity zero. The velocity byte contained within the note-off message is ignored.

Set grid LEDs

Hex version	90h, *Key*, *Velocity*.
Decimal version	144, *Key*, *Velocity*.

A note-on message changes the state of a grid LED. *Key* is the MIDI note number, which determines the LED location. *Velocity* is used to set the LED colour. Launchpad can be configured to map its buttons to MIDI note messages in one of two ways. The differences between these mapping modes are covered later, and can be seen in Figures 1 and 2. The default mapping is the X-Y layout. In this mapping, locations are addressed as follows, with the origin being the square button at the top-left corner of the grid:

Hex version	*Key* = (10h x *Row*) + *Column*
Decimal version	*Key* = (16 x *Row*) + *Column*

The scene launch buttons (the round buttons with printed triangles) are column 8. Invalid column numbers (9 to 15) are also interpreted as column 8..

Velocity is determined 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 Launchpad's double-buffering feature. See the 'Control double-buffering' message and the Appendix for details about how this can be used.

Launchpad is able to set the brightness of green and red LEDs 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:

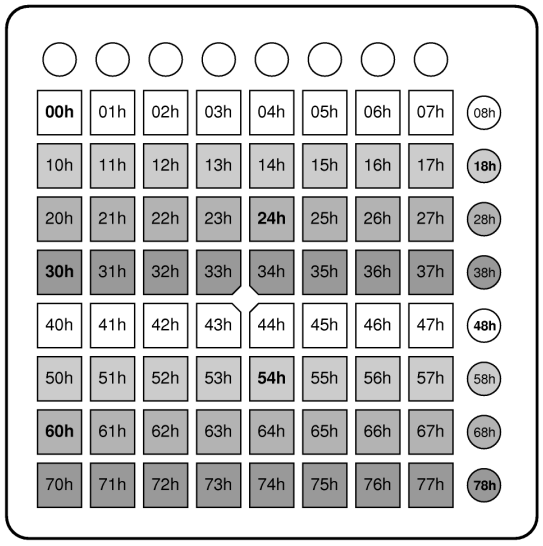
Hex	Decimal	Colour	Brightness
0Bh	11	Red	Full
3Bh	59	Amber	Full
3Ah	58	Yellow	Full
38h	56	Green	Full

The top row of round buttons, normally reserved for Automap and Live features, are accessed using controller change messages 68–6Fh. These are described elsewhere in this manual.

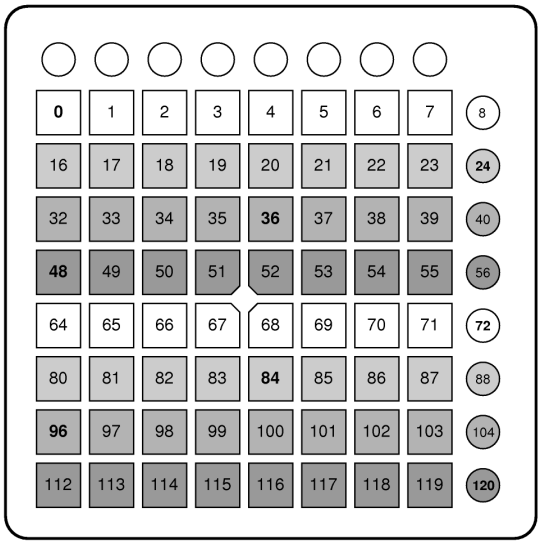
Figure 1. Launchpad in X-Y layout (mapping mode 1).
These diagrams express the same numbers in three different forms.

Every MIDI key code in bold text is a 'C'.
Grey shading is used to clarify the pattern in which the keys are arranged.

X-Y layout (Hex equivalent)



X-Y layout (Decimal equivalent)



X-Y layout (MIDI note equivalent)

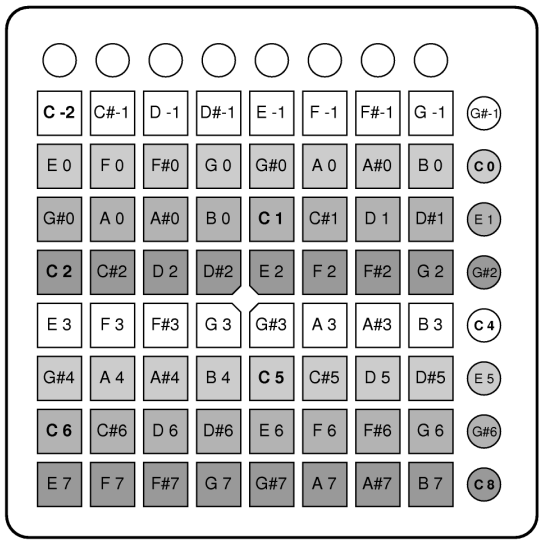
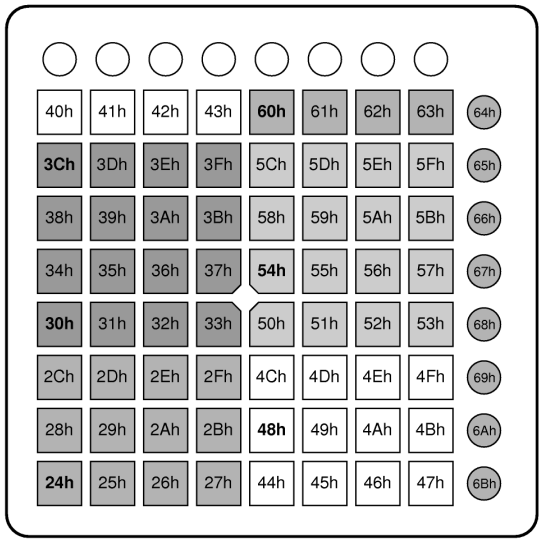


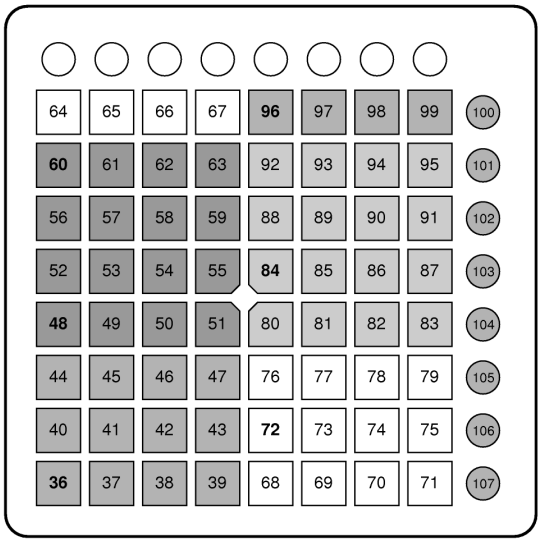
Figure 2. Launchpad in drum layout (mapping mode 2).
These diagrams express the same numbers in three different forms.

Every MIDI key code in bold text is a 'C'.
Grey shading is used to clarify the pattern in which the keys are arranged.

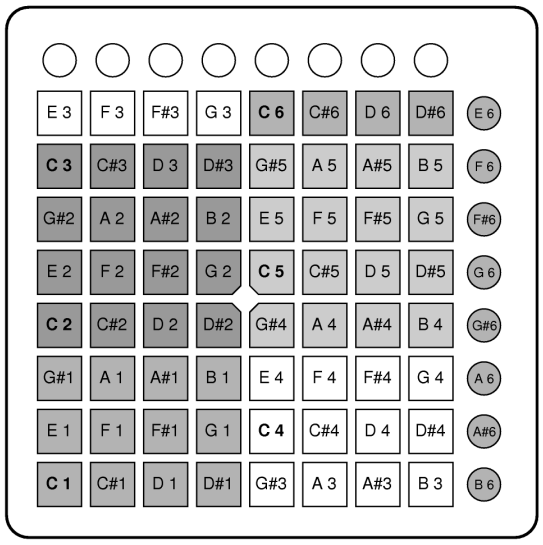
Drum rack layout (Hex equivalent)



Drum rack layout (Decimal equivalent)



Drum rack layout (MIDI note equivalent)



Reset Launchpad

Hex version	B0h, 00h, 00h.
Decimal version	176, 0, 0.

All LEDs are turned off, and the mapping mode, buffer settings, and duty cycle are reset to their default values.

Select the grid mapping mode

Hex version	B0h, 00h, 01-02h.
Decimal version	176, 0, 1-2.

This command affects the mapping of Launchpad buttons to MIDI key codes for messages in both directions. There are two possible mappings, selectable with the last byte of this message:

Mapping	**Meaning**
1	X-Y layout (the default).
2	Drum rack layout.

The X-Y layout is best for writing applications that use the Launchpad as a free grid, as it is easy to navigate a cursor around the 8x8 grid in any direction by simply adding or subtracting. The drum rack layout is better for situations when the Launchpad must deal with musical MIDI notes: six continuous octaves are available in this mode, and these are laid out in a regular pattern.

Figures 1 and 2 illustrate the button locations and MIDI note numbers in each mapping mode.

Control double-buffering

Hex version	B0h, 00h, 20-3Dh.
Decimal version	176, 0, 32-61.

See the Appendix for more information on double-buffering. The last byte is given 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 Launchpad is displayed instantaneously.

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

Turn on all LEDs

Hex version	B0h, 00h, 7D-7Fh.
Decimal version	176, 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 Launchpad* message for more information.

Set the duty cycle

```
Hex version                B0h, 1E-1Fh, Data.
Decimal version            176, 30-31, Data.
```

Launchpad controls the brightness of its LEDs by continually switching them on and off faster than the eye can see: a technique known as multiplexing. This command provides a way of altering the proportion of time for which the LEDs are on while they are in low- and medium-brightness modes. This proportion is known as the *duty cycle*.

Manipulating this is useful for fade effects, for adjusting contrast, and for creating custom palettes. The duty cycle is encoded in the controller number as well as the data byte, as follows:

```
Duty cycle = numerator / denominator

        where numerator   is a number between 1 and 16;
              denominator is a number between 3 and 18.
```

If *numerator* is less than 9, send B0h, 1Eh (176, 30), and then the following:

```
Hex version       Data = ( 10h x (numerator – 1) )
                       +  (denominator – 3)

Decimal version   Data = ( 16 x (numerator – 1) )
                       +  (denominator – 3)
```

Otherwise, send B0h, 1Fh (176, 31), and then the following:

```
Hex version       Data = ( 10h x (numerator – 9) )
                       +  (denominator – 3)

Decimal version   Data = ( 16 x (numerator – 9) )
                       +  (denominator – 3)
```

The medium-brightness LED duty cycle is always twice this number.

The default duty cycle is 1/5 (which would be set using B0h, 1Eh, 02h) meaning that low-brightness LEDs are on for only every fifth multiplex pass, and medium-brightness LEDs are on for two passes in every five. As another example, the low-brightness duty cycle could be set to to 2/7 by using B0h, 1Eh, 14h.

Generally, lower duty cycles (numbers closer to zero) will increase contrast between different brightness settings but will also increase flicker; higher ones will eliminate flicker, but will also reduce contrast. Note that using less simple ratios (such as 3/17 or 2/11) can also increase perceived flicker.

If you are particularly sensitive to strobing lights, please use this command with care when working with large areas of low-brightness LEDs: in particular, avoid duty cycles of 1/8 or less.

Set Automap/Live control LEDs

Hex version	B0h, 68-6Fh, *Data*.
Decimal version	176, 104-111, *Data*.

This command sets the LEDs under the top row of round buttons, normally reserved for Automap and Live features. The controller number determines the button's location: the leftmost button (cursor up/learn) is 68h (104 in decimal), and the controller number increases from left to right. The data byte sets the LED colour, and takes exactly the same format as the velocity byte in note-on messages.

Rapid LED update

Hex version	92h, *Velocity 1, Velocity 2,* *92h, Velocity 3, Velocity 4 ...*
Decimal version	146, *Velocity 1, Velocity 2,* *146, Velocity 3, Velocity 4 ...*

Sending a MIDI channel 3 note-on message enters a special LED update mode. All eighty LEDs may be set using only forty consecutive instructions, without having to send any key addresses.

Irrespective of the mapping chosen, this will update the 8x8 grid in left-to-right, top-to-bottom order, then the eight scene launch buttons in top-to-bottom order, and finally the eight Automap/Live buttons in left-to-right order (these are otherwise inaccessible using note-on messages). Overflowing data will be ignored.

To leave the mode, send a standard messsage beginning with 80h, 90h, or B0h. Sending another kind of message and then re-sending 92h will reset the cursor to the top left of the grid.

4 Launchpad-to-Computer messages

Grid button pressed

Hex version	90h, *Key*, *Velocity*.
Decimal version	144, *Key*, *Velocity*.

The *Key* is the key location, as described in the previous section and in Figures 1 and 2. A message is sent with velocity 7Fh (127 decimal) when a button is pressed. A second message is sent with velocity 0 when it is released.

Automap/Live button pressed

Hex version	B0h, 68-6Fh, *Data*.
Decimal version	176, 104-111, *Data*.

The leftmost button (cursor up/learn) is controller number 68h (104 decimal), and the controller number increases from left to right. A message is sent with velocity 7Fh (127 decimal) when a button is pressed down. A second message is sent with velocity 0 when it is released.

Appendix — LED double-buffering and flashing

The Launchpad 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 Launchpad 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 B0h, 00h, 31h (decimal 144, 0, 49) on start-up. This sets buffer 1 as the displayed buffer, and buffer 0 as the updating buffer. Launchpad will cease to show new LED data that is written to it.
2. Write LEDs to the Launchpad as usual, ensuring that the *Copy* and *Clear* bits are not set.
3. When this update is finished, send B0h, 00h, 34h (decimal 144, 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 new contents of buffer 0 will automatically be copied to buffer 1.
4. Write more LEDs to the Launchpad, with *Copy* and *Clear* bits set to zero.
5. When this update is finished, send B0h, 00h, 31h again (decimal 144, 0, 49). 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 B0h, 00h, 30h (decimal 144, 0, 48).

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

```
Hex version                B0h, 00h, 28h.
Decimal version            144, 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      B0h, 00h, 20h  (decimal version 144, 0, 32).
Turn flashing LEDs off     B0h, 00h, 21h  (decimal version 144, 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.