**Akai APC Mini MK2 Setup Guide for Ableton Live 12**

**Using Bome MIDI Translator Pro for Custom Mappings and LED Control**

This guide provides step-by-step instructions for setting up your Akai APC Mini MK2 with Ableton Live 12 through Bome MIDI Translator Pro, enabling custom mappings and LED feedback.

**Table of Contents**

1. [Initial Setup](https://claude.ai/chat/b75d5aaf-7992-4b47-a775-a062e8001a72#initial-setup)
2. [Configure Bome MIDI Translator Pro](https://claude.ai/chat/b75d5aaf-7992-4b47-a775-a062e8001a72#configure-bome-midi-translator-pro)
3. [Configure Ableton Live](https://claude.ai/chat/b75d5aaf-7992-4b47-a775-a062e8001a72#configure-ableton-live)
4. [Create Translators for LED Control](https://claude.ai/chat/b75d5aaf-7992-4b47-a775-a062e8001a72#create-translators-for-led-control)
5. [APC Mini MK2 MIDI Protocol Reference](https://claude.ai/chat/b75d5aaf-7992-4b47-a775-a062e8001a72#apc-mini-mk2-midi-protocol-reference)
6. [Troubleshooting](https://claude.ai/chat/b75d5aaf-7992-4b47-a775-a062e8001a72#troubleshooting)

**Initial Setup**

Before you begin:

* Ensure your APC Mini MK2 is connected to your computer via USB
* Install and launch Bome MIDI Translator Pro
* Have Ableton Live 12 installed and ready

**Configure Bome MIDI Translator Pro**

**Step 1: Set up MIDI Thru Paths**

1. Launch Bome MIDI Translator Pro and create a new project (or open your existing one)
2. Go to project properties (click on the project name at the top of the preset list)
3. Scroll down to the **MIDI Router** section
4. Create these two MIDI Thru paths:
   * From **APC MINI MK2** → **Bome MIDI Translator Virtual 1**
   * From **Bome MIDI Translator Virtual 1** → **APC MINI MK2**

**Step 2: Create Aliases (Optional but Recommended)**

1. In Project Properties, go to **MIDI Ports/Aliases**
2. Create an alias named "Ableton" for Bome MIDI Translator Virtual 1
   * Click "Create Alias" button
   * Name it "Ableton"
   * Select "Bome MIDI Translator Virtual 1" from the dropdown
3. Create an alias named "Controller" for your APC MINI MK2
   * Click "Create Alias" button
   * Name it "Controller"
   * Select "APC MINI MK2" from the dropdown

These aliases make your project more portable and easier to understand.

**Configure Ableton Live**

1. Open Ableton Live 12
2. Go to **Preferences** → **Link/MIDI**
3. In the MIDI Ports section:
   * Find **BMT 1** (Bome MIDI Translator Virtual 1) in the list
   * Enable "Track," "Sync," and "Remote" for both Input and Output
4. In the MIDI Remote section:
   * **Important**: Select "None" for control surface (do not select APC MINI MK2)
   * This preserves your custom mappings instead of loading Ableton's factory script
   * For the MIDI Input and Output, select **BMT 1**

This setup ensures all communication between Ableton and your controller passes through Bome MIDI Translator Pro without disturbing your custom mappings.

**Creating a Per-Track LED Feedback System**

Let's create a more sophisticated system that tracks clip states per track/column:

* Each of the 8 tracks can have its own active clip (lit red)
* When you trigger a new clip in the same track, only the previous clip in that track turns green
* Clips in other tracks remain unchanged

**Per-Track Clip Tracker**

1. In Bome MIDI Translator Pro, create a new preset named "APC Mini MK2 Lights"
2. Add a translator named "Per-Track Clip State Tracker"
3. Configure the **Incoming Action**:
   * Type: **MIDI Message**
   * Message Type: **Note On**
   * MIDI Channel: any (set to variable oo)
   * Note: any (set to variable pp)
   * Velocity: any (set to variable qq)
4. Configure the **Rules**:

// Only process pad notes (0-63)

if pp > 63 then Exit Rules, Skip Outgoing Action

// Determine which track (column) this pad belongs to

// Track number = pad number modulo 8

tt = pp % 8

// Store the previous playing clip in this track

// We'll use global variables g0-g7 to track active clips in each track

if tt == 0 then goto "Track0"

if tt == 1 then goto "Track1"

if tt == 2 then goto "Track2"

if tt == 3 then goto "Track3"

if tt == 4 then goto "Track4"

if tt == 5 then goto "Track5"

if tt == 6 then goto "Track6"

if tt == 7 then goto "Track7"

goto "ProcessClip" // Should never reach here

Label "Track0"

ga = g0 // Store previously active clip for this track

g0 = pp // Update active clip for this track

goto "ProcessClip"

Label "Track1"

ga = g1

g1 = pp

goto "ProcessClip"

Label "Track2"

ga = g2

g2 = pp

goto "ProcessClip"

Label "Track3"

ga = g3

g3 = pp

goto "ProcessClip"

Label "Track4"

ga = g4

g4 = pp

goto "ProcessClip"

Label "Track5"

ga = g5

g5 = pp

goto "ProcessClip"

Label "Track6"

ga = g6

g6 = pp

goto "ProcessClip"

Label "Track7"

ga = g7

g7 = pp

goto "ProcessClip"

Label "ProcessClip"

// If velocity > 0, this clip is playing

// If velocity = 0, this clip is stopped

if qq > 0 then goto "ClipPlaying"

goto "ClipStopped"

Label "ClipPlaying"

// Set this pad to red

rr = 5

Exit Rules, Execute Outgoing Action

Label "ClipStopped"

// Set this pad to green

rr = 21

// Clear this track's active clip if it's the one that stopped

// We need to use separate if statements - Bome doesn't support AND (&&) operators

if tt == 0 then goto "CheckTrack0"

if tt == 1 then goto "CheckTrack1"

if tt == 2 then goto "CheckTrack2"

if tt == 3 then goto "CheckTrack3"

if tt == 4 then goto "CheckTrack4"

if tt == 5 then goto "CheckTrack5"

if tt == 6 then goto "CheckTrack6"

if tt == 7 then goto "CheckTrack7"

goto "ExitRule"

Label "CheckTrack0"

if pp == g0 then g0 = 127

goto "ExitRule"

Label "CheckTrack1"

if pp == g1 then g1 = 127

goto "ExitRule"

Label "CheckTrack2"

if pp == g2 then g2 = 127

goto "ExitRule"

Label "CheckTrack3"

if pp == g3 then g3 = 127

goto "ExitRule"

Label "CheckTrack4"

if pp == g4 then g4 = 127

goto "ExitRule"

Label "CheckTrack5"

if pp == g5 then g5 = 127

goto "ExitRule"

Label "CheckTrack6"

if pp == g6 then g6 = 127

goto "ExitRule"

Label "CheckTrack7"

if pp == g7 then g7 = 127

goto "ExitRule"

Label "ExitRule"

Exit Rules, Execute Outgoing Action

1. Configure the **Outgoing Action**:
   * Type: **MIDI Message**
   * Message Type: **Note On**
   * MIDI Channel: **6** (for 100% brightness, value 96h)
   * Note: pp (same pad)
   * Velocity: rr (color value set in rules)

When Bome sends this message, it will construct the MIDI data as: 96 pp rr

* 96 = Note On, channel 6 (100% brightness solid lighting)
* pp = Note number (the pad we want to change)
* rr = Velocity (color value, 5 for red or 21 for green)

**Turn Previous Clip in Same Track Green**

We need a second translator to turn the previously active clip in the same track green:

1. Add another translator named "Reset Previous Clip in Track"
2. Configure the **Incoming Action**:
   * Type: **MIDI Message**
   * Message Type: **Note On**
   * MIDI Channel: any (set to variable oo)
   * Note: any (set to variable pp)
   * Velocity: any (set to variable qq)
3. Configure the **Rules**:

// Only process pad notes (0-63) with velocity > 0 (playing clips)

if pp > 63 then Exit Rules, Skip Outgoing Action

if qq == 0 then Exit Rules, Skip Outgoing Action

// Make sure previous clip isn't the same as current clip

if ga == pp then Exit Rules, Skip Outgoing Action

// Skip if no previous clip (value 127 is our "no clip" marker)

if ga == 127 then Exit Rules, Skip Outgoing Action

// Turn the previous clip green

ss = ga

tt = 21 // Green color

Exit Rules, Execute Outgoing Action

1. Configure the **Outgoing Action**:
   * Type: **MIDI Message**
   * Message Type: **Note On**
   * MIDI Channel: **6** (for 100% brightness, value 96h)
   * Note: ss (previous clip's pad)
   * Velocity: tt (green color)

When Bome sends this message, it will construct the MIDI data as: 96 ss tt

* 96 = Note On, channel 6 (100% brightness solid lighting)
* ss = Note number (the pad we want to change)
* tt = Velocity (color value, 21 for green)

**Initialization**

To initialize the global variables at project start:

1. Add a translator named "Initialize Track Variables"
2. Configure the **Incoming Action**:
   * Type: **Project**
   * Action: **The Project file is opened**
3. Configure the **Rules**:

// Initialize all track active clip variables to "no clip" (127)

g0 = 127

g1 = 127

g2 = 127

g3 = 127

g4 = 127

g5 = 127

g6 = 127

g7 = 127

Exit Rules, Skip Outgoing Action

This system will track the active clip in each of the 8 tracks independently. When you trigger a new clip in the same track, only the previous clip in that track turns green, while clips in other tracks remain unchanged.

1. Configure the **Outgoing Action**:
   * Type: MIDI
   * Message: 90 pp rr (Note On, original channel, same button, variable color)

**APC Mini MK2 MIDI Protocol Reference**

**Pad Matrix Values**

The APC Mini MK2 has an 8x8 grid of pads, with note numbers 0x00-0x3F:

0x38 0x39 0x3A 0x3B 0x3C 0x3D 0x3E 0x3F

0x30 0x31 0x32 0x33 0x34 0x35 0x36 0x37

0x28 0x29 0x2A 0x2B 0x2C 0x2D 0x2E 0x2F

0x20 0x21 0x22 0x23 0x24 0x25 0x26 0x27

0x18 0x19 0x1A 0x1B 0x1C 0x1D 0x1E 0x1F

0x10 0x11 0x12 0x13 0x14 0x15 0x16 0x17

0x08 0x09 0x0A 0x0B 0x0C 0x0D 0x0E 0x0F

0x00 0x01 0x02 0x03 0x04 0x05 0x06 0x07

**LED Behavior (MIDI Channel)**

The first byte of the MIDI message determines the LED behavior:

| **MIDI Channel** | **Byte 1 Value** | **Function** |
| --- | --- | --- |
| 0 | 90 | On 10% brightness |
| 1 | 91 | On 25% brightness |
| 2 | 92 | On 50% brightness |
| 3 | 93 | On 65% brightness |
| 4 | 94 | On 75% brightness |
| 5 | 95 | On 90% brightness |
| 6 | 96 | On 100% brightness |
| 7 | 97 | Pulsing 1/16 |
| 8 | 98 | Pulsing 1/8 |
| 9 | 99 | Pulsing 1/4 |
| 10 | 9A | Pulsing 1/2 |
| 11 | 9B | Blinking 1/24 |
| 12 | 9C | Blinking 1/16 |
| 13 | 9D | Blinking 1/8 |
| 14 | 9E | Blinking 1/4 |
| 15 | 9F | Blinking 1/2 |

**LED Colors (Velocity Values)**

The third byte (velocity) of the MIDI message determines the LED color:

| **Color** | **Velocity** | **Color** | **Velocity** | **Color** | **Velocity** |
| --- | --- | --- | --- | --- | --- |
| Off (black) | 0 | Green | 21 | Dark blue | 42 |
| Dim gray | 1 | Dark green | 22 | Very dark blue | 43 |
| Medium gray | 2 | Very dark green | 23 | Bright blue | 44 |
| White | 3 | Light green | 24 | Blue | 45 |
| Light red | 4 | Bright green | 25 | Dark blue | 46 |
| Red | 5 | Dark green | 26 | Very dark blue | 47 |
| Dark red | 6 | Very dark green | 27 | Light purple | 48 |
| Very dark red | 7 | Teal | 28 | Purple | 49 |
| Orange | 9 | Green blue | 29 | Dark purple | 50 |
| Dark orange | 10 | Dark teal | 30 | Very dark purple | 51 |
| Amber | 11 | Very dark teal | 31 | Pink | 52 |
| Yellow | 13 | Light cyan | 32 | Magenta | 53 |
| Dark yellow | 14 | Cyan | 33 | Dark pink | 54 |
| Very dark yellow | 15 | Dark cyan | 34 | Very dark pink | 55 |
| Light green | 16 | Very dark cyan | 35 | Light pink | 56 |
| Lime | 17 | Light blue | 36 | Rose | 57 |
| Dark lime | 18 | Sky blue | 37 | Dark rose | 58 |
| Very dark lime | 19 | Dark sky blue | 38 | Very dark rose | 59 |
| Bright green | 20 | Very dark sky blue | 39 | Orange red | 60 |

**Control Buttons and Faders**

| **Control Name** | **CC# / Note#** | **Channel** | **Type** | **Port** |
| --- | --- | --- | --- | --- |
| Track Button 1-8 | 0x64-0x6B | 0 | Note | 0 |
| Scene Launch 1-8 | 0x70-0x77 | 0 | Note | 0 |
| Shift | 0x7A | 0 | Note | 0 |
| Fader 1-8 | 0x30-0x37 | 0 | CC | 0 |
| Fader 9 (Master) | 0x38 | 0 | CC | 0 |

**Single LED Control**

For the non-pad buttons (Track buttons and Scene Launch buttons), use:

90 xx 01 // Turn on (xx is the button number)

90 xx 00 // Turn off

90 xx 02 // Blink

**Troubleshooting**

**No LED Feedback**

1. Check MIDI routing in Bome MIDI Translator Pro
   * Verify that MIDI thru paths are correctly set up
   * Monitor MIDI traffic with the Event Monitor
2. Verify Ableton settings
   * Make sure BMT 1 is selected for Track, Sync, and Remote
   * Confirm the correct MIDI Remote script is loaded
3. Test direct communication
   * Try sending direct MIDI messages to the controller through Bome
   * Example: 96 00 05 should light up the bottom-left pad in red

**Incorrect Colors**

1. Check your color values in the translator rules
   * Refer to the color chart in this guide
   * Test different velocity values to find the right colors
2. Verify MIDI channel for brightness/behavior
   * The first byte (9x) controls brightness and behavior
   * 96 is for 100% brightness

**Performance Issues**

If you experience latency or other performance issues:

1. Simplify your translators
   * Use fewer rules
   * Consider using the "Stop Processing" option for efficient processing
2. Check your computer resources
   * Close unnecessary applications
   * Increase buffer size in Ableton if needed

**Advanced Customization**

You can extend this setup to create more complex behaviors:

* Different pad colors based on track states
* Shift button functionality to access different layers
* Custom light shows for performance

Refer to the APC Mini MK2 MIDI Protocol documentation for more advanced options.

Created with guidance from Steve Caldwell, Bome Customer Care