3D Printed LED-Animation BMO

Created by Ruiz Brothers

https://learn.adafruit.com/3d-printed-led-animation-bmo

Last updated on 2023-08-29 02:29:31 PM EDT
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td>3</td>
</tr>
<tr>
<td>• Tools &amp; Supplies</td>
<td></td>
</tr>
<tr>
<td>• Parts</td>
<td></td>
</tr>
<tr>
<td>• Prerequisite Guides:</td>
<td></td>
</tr>
<tr>
<td>3D Printing</td>
<td>5</td>
</tr>
<tr>
<td>• Painting BMO</td>
<td></td>
</tr>
<tr>
<td>Circuit Diagram</td>
<td>6</td>
</tr>
<tr>
<td>• Gemma + 8x8 LED Matrix</td>
<td></td>
</tr>
<tr>
<td>• Prototyping</td>
<td></td>
</tr>
<tr>
<td>Arduino Code</td>
<td>7</td>
</tr>
<tr>
<td>• Writing Custom LED Animations</td>
<td></td>
</tr>
<tr>
<td>• Animation Tool for Adafruit 8x8 LED Matrix</td>
<td></td>
</tr>
<tr>
<td>CircuitPython Code</td>
<td>12</td>
</tr>
<tr>
<td>Assembling Circuit</td>
<td>15</td>
</tr>
<tr>
<td>• Setting up wires</td>
<td></td>
</tr>
<tr>
<td>• Mini 8x8 LED Matrix</td>
<td></td>
</tr>
<tr>
<td>• Wiring 8x8 LED Matrix</td>
<td></td>
</tr>
<tr>
<td>• Connecting 8x8 LED Matrix to GEMMA</td>
<td></td>
</tr>
<tr>
<td>• Switch/JST adapter</td>
<td></td>
</tr>
<tr>
<td>• Push Button</td>
<td></td>
</tr>
<tr>
<td>Assembling BMO</td>
<td>20</td>
</tr>
<tr>
<td>• Adding the 8x8 LED Matrix</td>
<td></td>
</tr>
<tr>
<td>• Adding Push Button</td>
<td></td>
</tr>
<tr>
<td>• Add Gemma</td>
<td></td>
</tr>
<tr>
<td>• Closing it Up</td>
<td></td>
</tr>
<tr>
<td>• Mounting JST/Switch adapter</td>
<td></td>
</tr>
<tr>
<td>• Add the Finishing Details</td>
<td></td>
</tr>
</tbody>
</table>
Overview

You can make your very own BMO with 3D Printing and DIY electronics. This guide will walk you through the steps of 3D printing the parts, soldering the components and assembling this delightful electronic companion.

The mini 8x8 LED matrix makes an awesome head for BMO, programmed to display several fun and playful facial expressions. Program your own animation to make new faces, letters or totally mathematical expressions ;-)}
This guide was written for the Gemma v2 board, but can be done with either the Gemma v2 or M0 Gemma. We recommend the Gemma M0 as it is easier to use and is more compatible with modern computers!

Tools & Supplies

- 3D Printer
- Soldering Iron
- Wire Strippers (http://adafru.it/527)
- 30 gauge wrapping wire (http://adafru.it/1446)

Parts

- Gemma M0 (http://adafru.it/1222) or Gemma v2 (http://adafru.it/1222)
- Lithium Polymer Battery (http://adafru.it/1570)
- Slide Switch (http://adafru.it/805)
- Button Switch (http://adafru.it/1119)

Prerequisite Guides:

- Introducing Gemma M0 guide (http://adafru.it/1222) or Introducing GEMMA guide (http://adafru.it/1222)
- Adafruit LED Backpacks (http://adafru.it/1570)
- Space Invaders Pendant (http://adafru.it/1570)
3D Printing

BMO's box enclosure, cover and limbs looks best printed in teal. A spool of teal colored ABS ranges from $30-50 depending on where you purchase it.

Below is a small list of places to get filament. Make sure your filaments diameter size matches your 3d printer. Most common size is 1.75mm and 3mm.

You can print BMO's buttons in different colored ABS or paint each piece. The buttons are rather small, so you will need to make sure your build plate is clean and nicely leveled. These parts are a bit difficult to print so make a few copies to make a batch of each. You can do this in MakerWare by copying and pasting them.

The tolerances in the holes are optimized for ABS but should work with PLA. You can use an x-acto knife in moderation to lose the tolerances to fit the components.

<table>
<thead>
<tr>
<th>bmo-box.stl</th>
<th>bmo-cover.stl</th>
<th>bmo-limbs.stl</th>
<th>bmo-green.stl</th>
<th>bmo-yellow.stl</th>
<th>bmo-blue.stl</th>
<th>bmp-green.stl</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABS 230c/120c</td>
<td>No Raft/Support</td>
<td>%15 infill</td>
<td>2 shells</td>
<td>0.2 layer height</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Takes around 2-3 hours to print all 10 pieces.

Download STLs

©Adafruit Industries
Painting BMO

If you 3D Print the buttons in different colored ABS, you only have to print BMO's letters on the sides.

Use a fine brush and even out the bristles by trimming them with sharp scissors.

Black acrylic paint works best on ABS prints. Apply small portions of paint and spread it on the surface of the raised letters on each side. Lightly dab on the edges to get a clean finished look. If you accidentally mark outside the letters, quickly rinse it down with water and try again. Let stand to dry for a few minutes.

Circuit Diagram
Gemma + 8x8 LED Matrix

Using the GEMMA in this build allows us to plug in a rechargeable battery directly into the JST port is directly on the board. GEMMA can be powered by USB but will need to have a battery power source for portability.

- VCC pin goes to VBAT
- GND pin goes to GND
- SCL pin goes to D2/A1
- SDA pin goes to D0

Prototyping

If your new to working with GEMMA, be sure to use a breadboard and jumper wires to prototype. Use alligator clips to clip jumper wires to GEMMA. The 8x8 LED matrix w/ backpack will need header + pins soldered to the breakouts. Check out the adafruits guide to LED backpacks for more details.

Arduino Code

The Arduino code presented below works well on GEMMA v2.. But if you have an M0 board you must use the CircuitPython code on the next page of this guide, no Arduino IDE required!
If this is your first time using Gemma, work through the Introducing Gemma guide first; you need to customize some settings in the Arduino IDE.

Be sure to check out Phil Burgess’s space invader pendant guide for getting the source code on to GEMMA properly.

Once you have your GEMMA/Trinket profiles installed in your Adafruit Arduino IDE, copy this code into a new sketch in Arduino and save it as matrix.ino.

```c
// SPDX-FileCopyrightText: 2014 Phil Burgess for Adafruit Industries
// SPDX-License-Identifier: MIT

// Trinket/Gemma + LED matrix backpack jewelry. Plays animated sequence on LED matrix. Press reset button to display again, or add optional momentary button between pin #1 and +V. THERE IS NO ANIMATION DATA IN THIS SOURCE FILE, you should rarely need to change anything here. EDIT anim.h INSTEAD.

#define BRIGHTNESS   12 // 0=min, 15=max
#define I2C_ADDR   0x70 // Edit if backpack A0/A1 jumpers set

#include <TinyWireM.h>
#include <avr/power.h>
#include <avr/sleep.h>
#include "bmo.h"       // Animation data is located here

static const uint8_t PROGMEM reorder[] = { // Column-reordering table
  0x00,0x40,0x20,0x60,0x10,0x50,0x30,0x70,0x08,0x48,0x28,0x68,0x18,0x58,0x38,0x78,
  0x04,0x44,0x24,0x64,0x14,0x54,0x34,0x74,0x0c,0x4c,0x2c,0x6c,0x1c,0x5c,0x3c,0x7c,
  0x02,0x42,0x22,0x62,0x12,0x52,0x32,0x72,0x0a,0x4a,0x2a,0x6a,0x1a,0x5a,0x3a,0x7a,
  0x06,0x46,0x26,0x66,0x16,0x56,0x36,0x76,0x0e,0x4e,0x2e,0x6e,0x1e,0x5e,0x3e,0x7e,
  0x01,0x41,0x21,0x61,0x11,0x51,0x31,0x71,0x09,0x49,0x29,0x69,0x19,0x59,0x39,0x79,
  0x05,0x45,0x25,0x65,0x15,0x55,0x35,0x75,0x0d,0x4d,0x2d,0x6d,0x1d,0x5d,0x3d,0x7d,
  0x03,0x43,0x23,0x63,0x13,0x53,0x33,0x73,0x0b,0x4b,0x2b,0x6b,0x1b,0x5b,0x3b,0x7b,
  0x07,0x47,0x27,0x67,0x17,0x57,0x37,0x77,0x0f,0x4f,0x2f,0x6f,0x1f,0x5f,0x3f,0x7f,
  0x00,0xc0,0xa0,0xe0,0x00,0x90,0xd0,0xb0,0x00,0x80,0xc8,0xa8,0xe8,0x00,0x70,0xd8,0xb8,0xf8,
  0x04,0xc4,0xa4,0xe4,0x04,0x94,0xd4,0xb4,0x04,0x8c,0xc8,0xac,0xe8,0x04,0x7c,0xdc,0xbc,0xfc,
  0x02,0xc2,0xa2,0xe2,0x02,0x92,0xd2,0xb2,0x02,0x8a,0xca,0xea,0xe9,0x0a,0xda,0xba,0xfa,
  0x06,0xc6,0xa6,0xe6,0x06,0x96,0xd6,0xb6,0x06,0xf6,0xe8,0xce,0xae,0xee,0xe9,0xe0,0xe0,0xe0,
  0x01,0xc1,0xa1,0xe1,0x01,0x91,0xd1,0xb1,0x01,0x89,0xca,0xe9,0x09,0x9a,0x9a,0xa9,0xa9,
  0x05,0xc5,0xa5,0xe5,0x05,0x95,0xd5,0xb5,0x05,0xf5,0xe8,0xc8,0xe8,0xe8,0xe8,0xe8,0xe8,
  0x03,0xc3,0xa3,0xe3,0x03,0x93,0xd3,0xb3,0x03,0xf3,0xe8,0xc8,0xc8,0xc8,0xc8,0xc8,
  0x07,0xc7,0xa7,0xe7,0x07,0x97,0xd7,0xb7,0x07,0xf7,0xf8,0xcf,0xcf,0xcf,0xcf,0xcf,
  0xff
};

void ledCmd(uint8_t x) { // Issue command to LED backpack driver
  TinyWireM.beginTransmission(I2C_ADDR);
  TinyWireM.write(x);
  TinyWireM.endTransmission();
}

void clear(void) { // Clear display buffer
  TinyWireM.beginTransmission(I2C_ADDR);
  for(uint8_t i=0; i<17; i++) TinyWireM.write(0);
  TinyWireM.endTransmission();
}

void setup() {
  power_timer1_disable(); // Disable unused peripherals
}
```

©Adafruit Industries
Create a new window and paste in the BMO animation code and save it as bmo.h in the same directory as the matrix sketch.

You may have to remove the .ino extension that gets added to the bmo.h file making it bmo.h.ino, remove that last .ino appending to just .h. Close the IDE and reopen it. The matrix sketch should show the bmo.h file in a tab.

Goto the tools file menu and select the Adafruit Gemma 8MHz profile under the Board drop down menu. Select USBTinyISP under the Programer drop down in the tool file menu.

Plug in a mini USB cable to the Gemma and connect it to your computer. Wait until you see a red blinkling LED on the Gemma and then hit the left arrow icon button to upload the code.
If you get a message in the IDE "Done Uploading" the code successfully uploaded to the GEMMA!

```
// SPDX-FileCopyrightText: 2014 Phil Burgess for Adafruit Industries
// SPDX-License-Identifier: MIT

// Animation data for Trinket/Gemma + LED matrix backpack jewelry.
#define REPS 255 // Number of times to repeat the animation loop (1-255)
const uint8_t PROGMEM anim[] = {
  B11111111, // 1 frame
  B10011001,
  B10011001,
  B11111111,
  B10000001,
  B11000011,
  B11100111,
  B11111111,
  25, // 0.25 second delay
  B11111111, // 2 frame
  B10011001,
  B10011001,
  B11111111,
  B10111101,
  B10111101,
  B11000011,
  B11111111,
  25, // 0.25 second delay
  B11111111, // 3 frame
  B10011001,
  B10011001,
  B11111111,
  B10000001,
  B11000011,
  B11100111,
  B11111111,
  25, // 0.25 second delay
  B11111111, // 4 frame
  B10011001,
  B10011001,
  B11111111,
  B10111101,
  B10111101,
  B11000011,
  B11111111,
  25, // 0.25 second delay
  B11111111, // 5 frame
  B10011001,
  B10011001,
  B11111111,
  B11111111,
  B10000001,
  B11111111,
  25, // 0.25 second delay
  B11111111, // 6 frame
  B10011001,
  B10011001,
  B11111111,
  B11100111,
  B11011011,
```
Writing Custom LED Animations

Check out Phil Burgess's Space invader guide for a great break down on how to write your own LED animations.

Animation Tool for Adafruit 8x8 LED Matrix

Check out this awesome website for building letter based LED animations. You simply click on letters to build an animation which can then be outputted to animation data for the Trinket/Gemma. It's awesome!
CircuitPython Code

GEMMA M0 boards can run CircuitPython — a different approach to programming compared to Arduino sketches. In fact, CircuitPython comes factory pre-loaded on GEMMA M0. If you've overwritten it with an Arduino sketch, or just want to learn the basics of setting up and using CircuitPython, this is explained in the Adafruit GEMMA M0 guide.

Below is CircuitPython code that works similarly (though not exactly the same) as the Arduino sketch shown on a prior page. To use this, plug the GEMMA M0 into USB...it should show up on your computer as a small flash drive...then edit the file “code.py” with your text editor of choice. Select and copy the code below and paste it into that file, entirely replacing its contents (don't mix it in with lingering bits of old code). When you save the file, the code should start running almost immediately (if not, see notes at the bottom of this page).

If GEMMA M0 doesn't show up as a drive, follow the GEMMA M0 guide link above to prepare the board for CircuitPython.

This code requires two additional libraries be installed:

1. adafruit_circuitpython_ht16k33
2. adafruit_bus_device

These directions are specific to the “M0” GEMMA board. The GEMMA v2 with an 8-bit AVR microcontroller doesn’t run CircuitPython...for those boards, use the Arduino sketch on the “Arduino code” page of this guide.
If you've just reloaded the board with CircuitPython, create the “lib” directory and then download the Adafruit CircuitPython Bundle.

Due to pin differences on the Gemma M0 versus the Gemma v2 we are using D1 as the optional touch based reset. The onboard Gemma M0 switch and reset button will also reset the animation.

```python
# SPDX-FileCopyrightText: 2014 Phil Burgess for Adafruit Industries
# SPDX-FileCopyrightText: 2018 Phil Burgess for Adafruit Industries
# SPDX-License-Identifier: MIT
# Trinket/Gemma + LED matrix backpack jewelry. Plays animated sequence on LED matrix. Press reset button to display again.

import time
import adafruit_ht16k33.matrix
import board
import busio as io
import touchio

touch = touchio.TouchIn(board.D1)
i2c = io.I2C(board.SCL, board.SDA)
matrix = adafruit_ht16k33.matrix.Matrix8x8(i2c)

# pixels initializers
x_pix = y_pix = 8
x = y = 0
matrix.fill(0)
matrix.show()

# seconds to pause between frames
frame_delay = [.25, .25, .25, .25, .25, .25, .25, .25, .25, .25]

# counter for animation frames
frame_count = 0

# repeat entire animation multiple times
reps = 255
rep_count = reps

# animation bitmaps
animation = [
    # frame 1
    [[1, 1, 1, 1, 1, 1, 1, 1], [1, 0, 0, 1, 1, 0, 0, 1],
     [1, 0, 0, 1, 1, 0, 0, 1], [1, 1, 1, 1, 1, 1, 1, 1],
     [1, 0, 0, 0, 0, 0, 0, 1], [1, 1, 0, 0, 0, 0, 1, 1],
     [1, 1, 1, 0, 0, 1, 1, 1], [1, 1, 1, 1, 1, 1, 1, 1]],
    # frame 2
    [[1, 1, 1, 1, 1, 1, 1, 1], [1, 0, 0, 1, 1, 0, 0, 1],
     [1, 0, 0, 1, 1, 0, 0, 1], [1, 1, 1, 1, 1, 1, 1, 1],
     [1, 0, 1, 1, 1, 1, 0, 1], [1, 0, 1, 1, 1, 1, 0, 1],
     [1, 1, 0, 0, 0, 0, 1, 1], [1, 1, 1, 1, 1, 1, 1, 1]],
    # frame 3
    [[1, 1, 1, 1, 1, 1, 1, 1], [1, 0, 0, 1, 1, 0, 0, 1],
     [1, 0, 0, 1, 1, 0, 0, 1], [1, 1, 1, 1, 1, 1, 1, 1],
     [1, 0, 0, 0, 0, 0, 0, 1], [1, 1, 0, 0, 0, 0, 1, 1],
     [1, 1, 1, 0, 0, 1, 1, 1], [1, 1, 1, 1, 1, 1, 1, 1]],
    # frame 4
    [[1, 1, 1, 1, 1, 1, 1, 1], [1, 0, 0, 1, 1, 0, 0, 1],
     [1, 0, 0, 1, 1, 0, 0, 1], [1, 1, 1, 1, 1, 1, 1, 1],
     [1, 0, 1, 1, 1, 1, 0, 1], [1, 0, 1, 1, 1, 1, 0, 1],
     [1, 1, 0, 0, 0, 0, 1, 1], [1, 0, 1, 1, 1, 1, 0, 1]],
]
```
[1, 1, 0, 0, 0, 0, 1, 1], [1, 1, 1, 1, 1, 1, 1, 1]],
# frame 5
[[1, 1, 1, 1, 1, 1, 1, 1], [1, 0, 0, 1, 1, 0, 0, 1],
[1, 0, 0, 1, 1, 0, 0, 1], [1, 1, 1, 1, 1, 1, 1, 1],
[1, 0, 0, 0, 0, 0, 0, 1], [1, 1, 1, 1, 1, 1, 1, 1]],
# frame 6
[[1, 1, 1, 1, 1, 1, 1, 1], [1, 0, 0, 1, 1, 0, 0, 1],
[1, 0, 0, 1, 1, 0, 0, 1], [1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 0, 0, 1, 1, 1], [1, 1, 0, 1, 1, 0, 1, 1],
[1, 1, 1, 0, 0, 1, 1, 1], [1, 1, 1, 1, 1, 1, 1, 1]],
# frame 7
[[1, 1, 1, 1, 1, 1, 1, 1], [1, 0, 0, 1, 1, 0, 0, 1],
[0, 0, 0, 1, 1, 0, 0, 0], [1, 0, 0, 1, 1, 0, 0, 1],
[1, 1, 1, 0, 0, 1, 1, 1], [1, 1, 0, 1, 1, 0, 1, 1],
[1, 1, 1, 0, 0, 1, 1, 1], [1, 1, 1, 1, 1, 1, 1, 1]],
# frame 8
[[1, 1, 1, 1, 1, 1, 1, 1], [1, 1, 1, 1, 1, 1, 1, 1],
[0, 0, 0, 1, 1, 0, 0, 0], [1, 1, 1, 1, 1, 1, 1, 1],
[1, 1, 1, 0, 0, 1, 1, 1], [1, 1, 0, 1, 1, 0, 1, 1],
[1, 1, 1, 0, 0, 1, 1, 1], [1, 1, 1, 1, 1, 1, 1, 1]],
# frame 9
[[1, 1, 1, 1, 1, 1, 1, 1], [1, 0, 0, 1, 1, 0, 0, 1],
[0, 0, 0, 1, 1, 0, 0, 0], [1, 0, 0, 1, 1, 0, 0, 1],
[1, 1, 1, 0, 0, 1, 1, 1], [1, 1, 0, 1, 1, 0, 1, 1],
[1, 1, 1, 0, 0, 1, 1, 1], [1, 1, 1, 1, 1, 1, 1, 1]],
]

# run until we are out of animation frames
# use Gemma's built-in reset button or switch to restart
#
# populate matrix
while True:
    if frame_count < len(animation) and rep_count >= 0:
        for x in range(x_pix):
            for y in range(y_pix):
                matrix.pixel(x, y, animation[frame_count][x][y])

        # next animation frame
        frame_count += 1

        # show animation
        matrix.show()

        # pause for effect
        time.sleep(frame_delay[frame_count])
    else:
        matrix.fill(0)
        matrix.show()
        time.sleep(.1)

        # track repitions
        rep_count -= 1

        # play it again
        frame_count = 0

        # A0/D1 pin has been touched
        # reset animation
        if touch.value:
            frame_count = 0
            rep_count = reps
Assembling Circuit

Setting up wires

You will need to cut and strip 6 strands of 30 guage wrapping wire for connecting the button and 8x8 LED matrix to GEMMA. Measure the length of the strips to about 90mm. You can optionally add labels to each wire with a piece of masking tape. If your new to working with GEMMA, check out the intro guide.

Mini 8x8 LED Matrix

The 0.8' led matrix comes with an I2c Backpack that will need to be soldered to the matrix. Carefully slide the matrix into the backpack and flip it over. Triple check that the text is on the same side as the From Adafruit text. Be sure to check out the 8x8 LED matrix intro guide

Solder all 16 pins. Apply the soldering iron to the pin and hold it there for a few seconds. Add small amount of solder to heated pin, not the tip of the iron! Clip and trim the pins when you’re finished soldering.

Wiring 8x8 LED Matrix

Apply a small amount of solder to tin the 4 pins on the top of the 8x8 LED matrix backpack. Try to heat the back of the pin, and then applying the solder to the front of the pin.

Solder 4 strips of 30 gauge wrapping wire to the 4 pins on the backpack. Try heating the back and pushing the wire through once it's liquidy.
Connecting 8x8 LED Matrix to GEMMA

Use a third helping hand (http://adafruit.it/291) to hold your components in place while you solder. Tin the 6 pins on the GEMMA. Solder the 4 wires from the LED Matrix to the pins of the GEMMA. Below is a mini reference list for each pin.

- SLC to D2/A1
- SDA to D0
- GND to GND
- VCC to VBat

And here’s the GEMMA + 8x8 LED matrix circuit! At this point, you can plug in a lipo battery to the JST to power it on and test if it works. The mini 8x8 LED Matrix should have enough slack to move freely from the GEMMA.
Switch/JST adapter

For conveniently powering our circuit, we recommend creating a switch/JST adapter. Using a JST extension cable, we can splice it onto a slide switch. The male end of the JST connects to the GEMMA, while the female port connects to your rechargeable battery.

Tin the middle and any outside pin of the slide switch with a small amount of solder. Be sure to apply rosin to each pin in order for the solder to stick properly. Use heat shrink tubing (http://adafruit.it/344) to secure our soldered connections.

Snip (http://adafruit.it/152) off the extra pin in the switch. It could cause a short in our circuit once it's mounted in the box enclosure.
Push Button

Use a tactile push button for replaying the LED animations.
Snip off any two diagonal pins and tin them.

Before soldering, be sure to add a piece of heat shrink tubing for securing the connection after you solder.

Solder one piece of wrapping wire to each pin.

With the wires securely soldered, put the heat shrink tubing over the connects and heat it up to seal the connections.

Once the button is wired up, solder the two wires to the GEMMA. Order of pins doesn’t matter as long as one goes to 3V and the other to D1.

Pin A to 3V
Pin B to D1
Here's what the complete circuit looks like. The push button should also have enough slack for it to move around in the enclosure. Gemma acts as BMO's brains, while the LED matrix+backpack is his face. We can assume the battery is his heart ;-)  

The next page shows you how to put these electro guts into BMO's boxy box.  

Assembling BMO  

Be careful when moving the cables around! Be super cautious not to over flex/bend any of the wires, it could totally break and that sucks! But if any of them do, you can re-solder your connections.
Adding the 8x8 LED Matrix

Before adding the LED matrix, add a black piece of tape to cover up the slot right below BMO's face. It's just for decoration, it doesn't actually do anything but adds a great finishing detail to make BMO look more like BMO.

Position the 8x8 LED matrix 'upright' by making sure the pin outs are at the top and adafruit logo at the bottom. Insert the LED Matrix into box with the top going in first, and then the bottom. The LED matrix should loosely fit into the BMO's box enclosure. Ensure the wiring in the pinout are positioned to the top.
Adding Push Button

Gently move the wiring of the LED matrix away from the top clips.

Slightly bend one of the pins of the push button inward, so it does not touch any part of the LED Matrix.

Insert one side of the button into the clips near the top of the box. Align up the push button with the hole in the top middle. Apply pressure to the opposite side of the button while pushing the top of the box inward. This will make a slight bend allowing the push button to snap into place.
Add Gemma
Place GEMMA over the back of the box cover with the USB port aligned up to the bottom opening. Gently press GEMMA down into the cover so it stays put.

Add the JST/Switch adapter to the GEMMA. Make sure two cables are not too long or else it won't fit inside the box.
Closing it Up

You will need to neatly fit the GEMMA, lipo battery and the JST/switch adapter into the box. For an easier build, try using a 110mAh lithium polymer battery (the 150mAh is a little more tricky to fit into the box, but it can be done!). Place the battery in between the Gemma and led matrix.

If the cover doesn't quite close, you may need to reduce the length of the battery cable and/or the JST/switch adapter. With patience, trial and error, you’re box will nicely close shut with a little wiggle room.
Mounting JST/Switch adapter
Carefully adjust the cables and insert the switch through the cut out near the top of the box cover. If the wires are too long, you may need to cut off some slack so all the components fit inside the box.
Add the Finishing Details
If you have already, pop in the button pieces into the holes on the front of BMO's boxy box. If they don't fit, using an x-acto knife to loosen up the opening. Be careful, don't over loosen the holes or the limbs and buttons won't stay put! But if you do, you can super glue them into place. BMO's arms and legs should freely rotate.

Add a split ring to the cover and put it on a necklace to wear your new electronic pal for a 3d printed adventure time!