Mini Pinball Table with Gemma M0

Created by John Park

https://learn.adafruit.com/mini-pinball-table-with-gemma-m0

Last updated on 2023-08-29 03:26:05 PM EDT
# Table of Contents

## Overview
- Control Parts
- Table Materials
- Tools

## Code the Gemma M0 with CircuitPython
- Gemma M0
- Coding Options
- iCade Standard
- Keyboard Mapping
- Test Code
- Button Code
- Download the Project Bundle
- Library Pruning

## Connect the Arcade Buttons

## Assemble the Aluminum Table
- Cutlist
- Assembly
- Front Rail and Plunger Button
- Side Flipper Buttons
- Front Rail Connection to Side Rails
- Front Leg Attachment
- Rear Leg Attachment
- Rear Frame Supports
- Gemma M0 Mount

## Play Pinball!
Overview

It's a lot of fun to play virtual pinball on an iPad! My favorite game is Pinball Arcade which features excellent physics and a huge host of classic table reproductions, from "Aadams Family Pinball" to "Theatre of Magic" and more. But, playing with the virtual, screen-tapping controls isn't as satisfying as hitting real flipper buttons. Let's solve that!

This Mini Pinball Table is pretty straightforward to build, using slotted aluminum extrusion and hardware. And, the circuit to make it all go is really easy to make -- it requires no soldering, just plug wires and clip alligator clips to the buttons and a Gemma M0 microcontroller, which takes care of the rest!

Any game that supports the iCade protocol can be controlled with the Gemma M0, Trinket M0, Circuit Playground Express, or other boards that can output HID USB keyboard strokes. See next page for more on the iCade protocol.
Control Parts

Listed below are the parts you'll need to make the controller

1 x Adafruit Gemma M0
Miniature wearable electronics microcontroller platform

https://www.adafruit.com/product/3501

2 x Arcade Button - 30mm
You'll need two buttons for the flippers, get them in any colors you like!

https://www.adafruit.com/product/476
<table>
<thead>
<tr>
<th>Quantity</th>
<th>Item Description</th>
<th>URL</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 x</td>
<td>Arcade Button with LED - 30mm</td>
<td><a href="https://www.adafruit.com/product/3490">https://www.adafruit.com/product/3490</a></td>
<td>3490</td>
</tr>
<tr>
<td>1 x</td>
<td>Arcade/Button Quick-Connect Wire Pair - Set of 10 pairs</td>
<td><a href="https://www.adafruit.com/product/1152">https://www.adafruit.com/product/1152</a></td>
<td></td>
</tr>
<tr>
<td></td>
<td>For simple button wiring</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td>Small Alligator Clip to Male Jumper Wire Bundle - 12 Pieces</td>
<td><a href="https://www.adafruit.com/product/3225">https://www.adafruit.com/product/3225</a></td>
<td>3225</td>
</tr>
<tr>
<td></td>
<td>To make connections between the button wiring and the Gemma M0 -- no soldering required!</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td>Short Wire Alligator Clip Test Leads (set of 12)</td>
<td><a href="https://www.adafruit.com/product/1592">https://www.adafruit.com/product/1592</a></td>
<td></td>
</tr>
<tr>
<td></td>
<td>You'll use these to connect buttons to common ground</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 x</td>
<td>USB cable - 6&quot; A/MicroB</td>
<td><a href="https://www.adafruit.com/product/898">https://www.adafruit.com/product/898</a></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Micro USB cable to connect the Gemma M0 to the USB-to-Lightning adapter</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

These controller parts are all you'll need should you decide to build your own table using other materials, such as LEGO, cardboard, rubber bands and pencils, or who knows what!

Table Materials

These are the parts used to build the aluminum table:

3 x Slotted Aluminum Extrusion - 20mm x 20mm -
610mm long
Slotted aluminum extrusions

https://www.adafruit.com/product/1221

<table>
<thead>
<tr>
<th>Part Description</th>
<th>Quantity</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 x L-Plate for 2020 Aluminum Extrusion</td>
<td>1</td>
<td><a href="https://www.adafruit.com/product/1218">https://www.adafruit.com/product/1218</a></td>
</tr>
<tr>
<td>L-plates for corner attachments, flipper button holders</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 x Aluminum Extrusion Double Corner Brace Support</td>
<td>1</td>
<td><a href="https://www.adafruit.com/product/1259">https://www.adafruit.com/product/1259</a></td>
</tr>
<tr>
<td>Corner braces</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 x Plastic Hinge for 20x20 Aluminum Extrusion</td>
<td>1</td>
<td><a href="https://www.adafruit.com/product/1215">https://www.adafruit.com/product/1215</a></td>
</tr>
<tr>
<td>Hinge for mounting the Gemma M0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 x Button Hex Machine Screw - M4 thread - 10mm long</td>
<td>1</td>
<td><a href="https://www.adafruit.com/product/1159">https://www.adafruit.com/product/1159</a></td>
</tr>
<tr>
<td>- pack of 50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Screws to hold things together</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 x Aluminum Extrusion Slim T-Nut for 20x20 - M4</td>
<td>1</td>
<td><a href="https://www.adafruit.com/product/1157">https://www.adafruit.com/product/1157</a></td>
</tr>
<tr>
<td>Thread - pack of 50</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Easy to use T-nuts -- don’t need to slide in from the</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ends!</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 x Hex Key/Allen Wrench 2.5mm</td>
<td>1</td>
<td><a href="https://www.adafruit.com/product/1229">https://www.adafruit.com/product/1229</a></td>
</tr>
<tr>
<td>For driving the M4 screws</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 x Aluminum Extrusion End Caps - pack of 10</td>
<td>1</td>
<td><a href="https://www.adafruit.com/product/1156">https://www.adafruit.com/product/1156</a></td>
</tr>
<tr>
<td>These act as feet for a non-scratch finish on the legs</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Additional Parts

You'll also need:

- iPad or iPhone and a copy of The Pinball Arcade or other iCade compatible game
- an adapter for the iOS device to connect USB to the Lightning port, such as the Apple USB to Lightning Camera Adapter. (If you have an older iPad or iPhone you will need the 30-pin dock connector version of the camera kit)
Tools

The only tools you'll need are the 2.5mm hex driver for the screws, and a hack saw and miter box or other tool (band saw, miter saw with metal cutting blade, Dremel, die grinder, etc.) to cut the lengths of aluminum extrusion.

Most cuts are at a 90 degree angle, but the four feet ends on the legs will be cut at an 87 degree angle to give the table it's signature tilt.

Next, let's program the Gemma M0 to turn button presses into keyboard commands!

---

Code the Gemma M0 with CircuitPython

Gemma M0

The Gemma M0 is just right for this project for a few reasons. It's inexpensive, has three digital input pins to read the two flippers and the plunger, it can output HID USB keyboard strokes over USB, and it draws very little power, so the iPad and iPhone won't mind powering it! Plus, it has an RGB DotStar LED that can be used as a multi-color indicator!
Coding Options

The Gemma M0 can currently be coded with either the Arduino IDE or CircuitPython. For this project, we'll use CircuitPython. It's very simple to set up -- all you need to do is plug it into your computer via USB and it shows up as a USB thumb drive. Then, you can write code in any text editor and save the file to the Gemma M0. That's all it takes, it immediately runs the code. Talk about fast iteration!

iCade Standard

Since the iPad doesn't have a game controller port on it like a video game console, game developers have come up with a few clever ways to interface physical controls with the device. One such standard is the iCade protocol, made by ION Audio for their line of iPad desktop arcade cabinets and controllers.
The iCade standard interfaces with iOS as an HID keyboard, and is typically paired over Bluetooth. However, plugging in a wired HID keyboard works just as well, which makes it very straightforward to build your own controller!

The Gemma M0 is going to act as a USB HID keyboard so that it can "type" the keystrokes that the iCade standard uses.
Here's what the button mapping looks like for the standard iCade:

![Button Commands Diagram]

### Keyboard Mapping

The first test I performed was to figure out the key mappings needed. I did so by pairing a Bluetooth keyboard to the iPad, launching Pinball Arcade, and setting the controller type to iCade in the game's settings. Then, I typed the key pairs seen in the diagram above until I knew which key combos did what.

The mapping for pinball looks like this:
Test Code

The next test I did was to have the Gemma M0 press the flipper buttons by telling it to "type" the "LVHR" combo over and over. Here's how you can make the Gemma M0 type the keystrokes.

First, make sure you've followed the basic guide to setting up the Gemma M0 ( ). Once you've successfully run the basic Blinky code example ( ) on your Gemma M0, move on to the next step.

If nothing happens when you plug in your Gemma M0, make sure the built-in ON/OFF switch is turned on!

The Gemma M0 ships with all of the libraries we'll need (HID library and DotStar library) already installed on the board, so we can immediately start coding this example!

With the Gemma M0 plugged into your computer, open the CIRCUTPY drive that shows up, and edit the code.py (or main.py, either will work) file in your text editor.

Copy and paste the code below into your text editor and then save the file, overwriting the existing code.py file on the root of the Gemma M0. Be careful where your cursor is when you save, since the program will cause the Gemma M0 to start spewing the letters "lhvr" as if you were banging on your keyboard!

If you want to test it out on your pinball game, go ahead and plug the Gemma M0 into the iPad using the USB to Lightning adapter. The iPad will autodetect the device and open a camera import window. You can ignore this and switch to your iCade compatible pinball game, such as Pinball Arcade. When you start a new game, it will
immediately begin flipping the flippers for you automatically! Launch a ball onto the field with the on-screen plunger and watch it play by itself for a bit!

**Button Code**

Now it's time to rewrite the code so that the two flippers and the plunger aren't haunted by the ghost of Tommy, but instead, respond to your arcade buttons.

**Download the Project Bundle**

Your project will use a specific set of CircuitPython libraries and the code.py file. In order to get the libraries you need, click on the Download Project Bundle link below, and uncompress the .zip file.

Drag the contents of the uncompressed bundle directory onto your Gemma M0 board's CIRCUITPY drive, with the exception of the adafruit_hid library (see next section on that) replacing any existing files or directories with the same names, and adding any new ones that are necessary.

---

### Library Pruning

You may run out of space on the Gemma M0 if you copy the entire adafruit_hid library over -- but you only need a few files, as shown on the left.

Delete unnecessary files from that folder before copying it over to the Gemma M0.

```
# SPDX-FileCopyrightText: 2022 john park & tod kurt for Adafruit Industries
# SPDX-License-Identifier: MIT
# Gemma IO demo - Keyboard emu
# iCade Pinball Edition

import board
from digitalio import DigitalInOut, Pull
from adafruit_debouncer import Debouncer
import usb_hid
from adafruit_hid.keyboard import Keyboard
from adafruit_hid.keycode import Keycode

# Allows three buttons on a Gemma M0 to control iCade standard Pinball Arcade game on iOS using USB to Lightning "camera connector"

# iCade keyboard mappings
```
# define buttons
num_keys = 3
pins = (board.D0, board.D1, board.D2)

# The keycode pair sent for each button:
# D0 is left flipper - iCade key sequence (hold, release) is "hr"
# D1 is right flipper - iCade key sequence (hold, release) is "lv"
# D2 is plunger - iCade key sequence (hold, release) is "xz"

for pin in pins:
    tmp_pin = DigitalInOut(pin)
    tmp_pin.pull = Pull.UP
    keys.append(Debouncer(tmp_pin))

keymap_pressed = {
    0: ("Left Paddle", [Keycode.H]),
    1: ("Right Paddle", [Keycode.L]),
    2: ("Plunger", [Keycode.X])
}

keymap_released = {
    0: ("Left Paddle", [Keycode.R]),
    1: ("Right Paddle", [Keycode.V]),
    2: ("Plunger", [Keycode.Z])
}

# the keyboard object
kbd = Keyboard(usb_hid.devices)

print("Welcome to keypad")
print("keymap:")
for k in range(num_keys):
    print("\t", (keymap_pressed[k][0])
print("Waiting for button presses")

while True:
    for i in range(num_keys):
        keys[i].update()
        if keys[i].fell:
            print(keymap_pressed[i][0])
            kbd.send(*keymap_pressed[i][1])
        if keys[i].rose:
            print(keymap_released[i][0])
            kbd.send(*keymap_released[i][1])

Next, we'll connect the arcade buttons to the Gemma M0.
Connect the Arcade Buttons

Using the diagram above, you can see how the three buttons will be connected. Since all buttons need to share a common ground with the Gemma M0, but we only have one available GND pad, we'll run ground to a terminal on one button (it doesn't matter which terminal) and then connect the rest of the buttons to that terminal.

You can try this arrangement out right away with the short alligator clip leads, or use the longer arcade button quick connect wire pairs and alligator clip to male jumper wires for the greater length we'll need in the final table as shown below.
To begin, connect the terminals of each wire pair to the lugs on the buttons.

Note: on the larger, threaded button with internal LED (which we won't be using), we'll connect only to the switch lugs which come out of the grey switch box, not the LED pair set deeper into the button housing.

Next, connect the ground wires as shown here using the black short alligator clip leads.
Connect the four wires extensions (black to GND, yellow to D1, blue to D2, and white to D0) to the alligator clip leads with male pins as shown here, being careful to connect to the correct leads, not the ground sides which will remain unused.

This is how the final wiring will look.
Plug in the Gemma M0 to power, and you can test out the buttons. The built-in DotStar is an RGB LED, and it make for a pretty handy indicator light. The code will light the DotStar to the same color as the button pressed as a handy way to check that everything is wired properly. Plus, it adds a bit of a fun light show while you're playing!

Now, you can try it out on your iPad (or iPhone as shown here). Plug the Gemma M0 into the short USB cable and then into the USB-to-Lightning adapter and into the iOS device. The camera import screen may pop up, you can ignore it and hit the home button. Then launch the Notes app and press the buttons to type the keystrokes.
OK, enough anticipation, launch the Pinball Arcade game and in the settings, click the iCade button until it reads "iCade:iPad". Then, launch a table, and test out your buttons!
Assemble the Aluminum Table

Now we get to build the table! This part is fun, sort of like building a kit, except you’ll need to do a little bit of cutting to create the proper lengths of aluminum extrusion.
Cutlist

First, take one of the lengths of aluminum extrusion and measure, mark, and cut the two frame sides from it, each at 300mm.
Measure, then scribe a mark into the aluminum with a sharp tool or knife.
Clamp the work into your miter box or fence of a miter saw
You can use a lubricant to keep the blade and work cool
Carefully cut the piece
File off any rough edges if needed
Next, cut the frame front and rear lengths to 197mm, also from a single 610mm length of stock.

This will leave enough remaining material to cut one of the rear legs, as well, which is 127mm, however, this one will have an 3 degree angled cut for the foot. So, set your miter angle to 3 degrees with the blade cutting the longest leg of the angle to 127mm.

The aluminum extrusion comes in 610mm lengths, so even with a bit of material lost to saw blade kerf, you can cut both sides from one piece.

Here we can see the angled cut next to a straight cut.
You can place the plastic end caps on the angled feet as well as the rear frame lengths.

Make the remaining cuts in the third piece of stock to create the other 127mm rear leg and both of the 102mm front legs. All legs will have the 3 degree cut to one end.

Assembly

Now we get to put it all together! Follow the photo sections below as your guide.
Front Rail and Plunger Button

The plunger button will connect to the front rail using two double corner brace supports.
Screw in two T-nuts on each bracket as shown here.
Loosely attach them to the front rail, then fit them tightly to the plunger button, with the button's plastic nut helping hold it in place.
Tighten down all four screws.
Side Flipper Buttons

Screw in two L-plates to the left rail making space for the left flipper button (Note the two extra screws in the forward plate which will be used to connect the front left leg later)
Overhang the front-most plate by 20mm, enough to over the front rail later
Hold the plates firmly against the button's catch tabs and then tighten the screws
Repeat for the right side flipper button
Prepare two more L-plates with screws and T-nuts. Connect the front rail to the right rail as shown. Make sure everything is nice and square, then connect the front rail to the left rail as shown.
Front Leg Attachment

The front legs slide into the two T-nuts that were already prepped on the L-plates. Make sure the foot angles are correct -- the table will have a forward lean to it. Set the legs up square to the undersides of the side rails, then screw down the screws.

Repeat for both sides.
Rear Leg Attachment

Prep two double corner brace supports, each with four screws and T-nuts.

Attach one of the brackets to the underside of the right side rail as shown, leaving 20mm of space for the leg to sit flush.

Again, being mindful of the foot angle that will allow for the table to lean properly, connect the right rear leg.

Tighten the screws, making sure everything's square.

Repeat these steps for the other rear leg.
Rear Frame Supports

Use two more L-plates to form supports that will hold the rear frame rail. This will allow for an easier time inserting and removing the iPad.

Note the position of the single screw and T-nut inserted from the underside of each plate; this will be used to connect the rear frame rail later (it does not need to be inserted just yet, and doesn't appear in all photos).
Gemma M0 Mount

Connect the Gemma M0 as shown to the plastic hinge using M2.5 nylon screws and standoffs or nuts. The best orientation for screw clearance is to have the Gemma M0 on the side of the plastic hinge with the smaller hole diameter, not the counter-sink side. Attach the plastic hinge/mount to the top rail with two screws and T-nuts.
Insert iPad and Close it Up

Slide the iPad into the frame, keeping the Lighting port at the top/rear of the table
Use two screws and T-nuts from the underside of the L-plates to attach the rear rail/Gemma M0 mount
You're just about ready to play!
Play Pinball!

Go ahead and re-connect the Gemma M0 to the short USB cable and USB-to-Lightning adapter, make sure the Gemma M0 is turned on, then fire up your pinball game!

There's enough space to slide the iPad up or down as needed to access the power button, and you can adjust the volume with the onscreen controls, since the physical volume buttons aren't accessible.

Here's the live stream video for more details: