Table of Contents

Overview 3
• 3D Printed Chain-able case
• Parts

Circuit Diagram 6
• Wired Connections
• Big Switches

CircuitPython 7
• CircuitPython Quickstart
• Safe Mode
• Flash Resetting UF2

Code 11
• Upload the Code and Libraries to the KBRP2040

3D Printing 13
• Parts List
• Slicing Parts

Assemble 14
• Solder wires
• Route wires
• Mount Big Switch
• Attaching multiple case
• Connect Big Switch
• Optional LED
Overview

Build a Giant MacroPad with Big Key Switches, CircuitPython and the KB2040!

Regular key switches are fairly small in size but these novelty ones are about 64 times as big!

This is powered by the KB2040 Kee Boar Driver running CircuitPython. It’s an Arduino Pro micro-shaped board that’s perfect for keyboard projects.

It features the RP2040 with 8MB of flash, 20 GPIO and STEMMA QT. It’s also got an on-board NeoPixel and a USB-C type connector.

You can set this up with your favorite macros and make yourself a jumbo keypad that’s super chunky and really fun to use.
3D Printed Chain-able case

We designed and 3D printed a case that allows you to daisy chain them together.

The KB2040 snap fits into the bottom cover and is held in place with little tabs.

Cables are routed through the holes on the side of the case so they can be connected together.

The bottom snap fits into the case with a cutout to access the board’s USB-C port.

Parts

The KB2040 is powered by a USB-C cable. It press fits into the bottom cover and is held in place with little tabs.

We used a 10mm LED to illuminate the Big Switch.

The Big Switch press fits into the printed case.

A lid attaches into the 3D printed case.
Adafruit KB2040 - RP2040 Kee Boar Driver
A wild Kee Boar appears! It’s a shiny KB2040! An Arduino Pro Micro-shaped board for Keebs with RP2040. (#keeblife 4 evah) A lot of folks like using Adafruit...
https://www.adafruit.com/product/5302

Kailh Big Mechanical Key Switch - Clicky Pale Blue - 1 Piece
The Kailh Big Switch Series comes to Adafruit! These switches are 4 times bigger on each side than a normal MX switch. That comes up to 64...
https://www.adafruit.com/product/5307

Kailh Big Mechanical Key Switch - Tactile Burnt Orange - 1 Piece
Kailh Big Switch Series comes to Adafruit! These switches are 4 times bigger on each side than a normal MX switch. That comes up to 64...
https://www.adafruit.com/product/5306

Kailh Big Mechanical Key Switch - Linear Dark Yellow - 1 Piece
The Kailh Big Switch Series comes to Adafruit! These switches are 4 times bigger on each side than a normal MX switch. That comes up to 64...
https://www.adafruit.com/product/5305
Diffused 10mm LED Pack - 5 LEDs each in 5 Colors - 25 Pack
Need some chunky indicators? We are big fans of these diffused LEDs. They are fairly bright, so they can be seen in daytime, and from any angle. They go easily into a breadboard and...
https://www.adafruit.com/product/4204

USB Type C Cable with Data/Charge Switch
Perhaps your smart phone, tablet, or even your Raspberry Pi 4, charges off of USB, but can you really trust that charging station at the airport? Now you can control when to allow...
https://www.adafruit.com/product/4696

Circuit Diagram
The diagram below provides a visual reference for wiring of the components. This diagram was created using the software package Fritzing ().

Use Adafruit's Fritzing parts library to create circuit diagrams for your projects. Download the library or just grab individual parts. Get the library and parts from GitHub - Adafruit Fritzing Parts ().
Wired Connections

The KB2040 is powered by a USB cable.

Big Switches

Each Big Switch is connected to the KB2040. The keys are illuminated by a 10mm LED.

- Switch 1 to Pin 2 on KB2040
- LED 1 to Pin 3 on KB2040
- Switch 2 to Pin 4 on KB2040
- LED 2 to Pin 5 on KB2040
- Switch 3 to Pin 6 on KB2040
- LED 3 to Pin 7 on KB2040
- All Grounds are shared between the three Pins on the KB2040

CircuitPython

[CircuitPython](https://circuitpython.org/) is a derivative of [MicroPython](https://micropython.org/) designed to simplify experimentation and education on low-cost microcontrollers. It makes it easier than ever to get prototyping by requiring no upfront desktop software downloads. Simply copy and edit files on the CIRCUITPY drive to iterate.
CircuitPython Quickstart

Follow this step-by-step to quickly get CircuitPython running on your board.

Download the latest version of CircuitPython for this board via circuitpython.org

Click the link above to download the latest CircuitPython UF2 file.

Save it wherever is convenient for you.

To enter the bootloader, hold down the BOOT/BOOTSEL button (highlighted in red above), and while continuing to hold it (don't let go!), press and release the reset button (highlighted in blue above). Continue to hold the BOOT/BOOTSEL button until the RPI-RP2 drive appears!

If the drive does not appear, release all the buttons, and then repeat the process above.
You can also start with your board unplugged from USB, press and hold the BOOTSEL button (highlighted in red above), continue to hold it while plugging it into USB, and wait for the drive to appear before releasing the button.

A lot of people end up using charge-only USB cables and it is very frustrating! Make sure you have a USB cable you know is good for data sync.

You will see a new disk drive appear called RPI-RP2.

Drag the adafruit_circuitpython_etc.uf2 file to RPI-RP2.

The RPI-RP2 drive will disappear and a new disk drive called CIRCUITPY will appear.

That's it, you're done! :)

©Adafruit Industries
Safe Mode

You want to edit your code.py or modify the files on your CIRCUITPY drive, but find that you can’t. Perhaps your board has gotten into a state where CIRCUITPY is read-only. You may have turned off the CIRCUITPY drive altogether. Whatever the reason, safe mode can help.

Safe mode in CircuitPython does not run any user code on startup, and disables auto-reload. This means a few things. First, safe mode bypasses any code in boot.py (where you can set CIRCUITPY read-only or turn it off completely). Second, it does not run the code in code.py. And finally, it does not automatically soft-reload when data is written to the CIRCUITPY drive.

Therefore, whatever you may have done to put your board in a non-interactive state, safe mode gives you the opportunity to correct it without losing all of the data on the CIRCUITPY drive.

Entering Safe Mode

To enter safe mode when using CircuitPython, plug in your board or hit reset (highlighted in red above). Immediately after the board starts up or resets, it waits 1000ms. On some boards, the onboard status LED (highlighted in green above) will blink yellow during that time. If you press reset during that 1000ms, the board will start up in safe mode. It can be difficult to react to the yellow LED, so you may want to think of it simply as a slow double click of the reset button. (Remember, a fast double click of reset enters the bootloader.)

In Safe Mode

If you successfully enter safe mode on CircuitPython, the LED will intermittently blink yellow three times.

If you connect to the serial console, you’ll find the following message.

```
Auto-reload is off.
Running in safe mode! Not running saved code.
CircuitPython is in safe mode because you pressed the reset button during boot. Press again to exit safe mode.
Press any key to enter the REPL. Use CTRL-D to reload.
```
You can now edit the contents of the CIRCUITPY drive. Remember, your code will not run until you press the reset button, or unplug and plug in your board, to get out of safe mode.

**Flash Resetting UF2**

If your board ever gets into a really weird state and doesn't even show up as a disk drive when installing CircuitPython, try loading this 'nuke' UF2 which will do a 'deep clean' on your Flash Memory. You will lose all the files on the board, but at least you'll be able to revive it! After loading this UF2, follow the steps above to re-install CircuitPython.

[Download flash erasing "nuke" UF2]

---

**Code**

Once you’ve finished setting up your KBRP2040 with CircuitPython, you can access the code and necessary libraries by downloading the Project Bundle.

To do this, click on the Download Project Bundle button in the window below. It will download as a zipped folder.

```python
# SPDX-FileCopyrightText: 2021 Collin Cunningham for Adafruit Industries
# SPDX-License-Identifier: MIT

import time
import board
import usb_hid
from adafruit_hid.keyboard import Keyboard
from adafruit_hid.keyboard_layout_us import KeyboardLayoutUS
from adafruit_hid.keycode import Keycode
from digitalio import DigitalInOut, Direction, Pull

# The pins connected to each switch/button
```
# The pins connected to each LED
ledpins = [board.D2, board.D6, board.D7]

# our array of button & LED objects
buttons = []
leds = []

# The keycode sent for each switch/button
buttonspressed = [False, False, False]
buttonspressedlast = [False, False, False]

# the keyboard object!
kbd = Keyboard(usb_hid.devices)
# we're americans :)
layout = KeyboardLayoutUS(kbd)

# make all button pin objects, make them inputs w/pullups
for pin in buttonpins:
    button = DigitalInOut(pin)
    button.direction = Direction.INPUT
    button.pull = Pull.UP
    buttons.append(button)

# make all LED objects, make them outputs
for pin in ledpins:
    led = DigitalInOut(pin)
    led.direction = Direction.OUTPUT
    leds.append(led)

# set up the status LED
statusled = DigitalInOut(board.D13)
statusled.direction = Direction.OUTPUT

print("Waiting for button presses")

def pressbutton(index):
    switch_led = leds[index]  # find the switch LED
    k = buttonkeys[index]  # get the corresp. keycode/str
    switch_led.value = True  # turn on LED
    kbd.press(k)  # send keycode

def releasebutton(index):
    switch_led = leds[index]  # find the switch LED
    k = buttonkeys[index]  # get the corresp. keycode/str
    switch_led.value = False  # turn on LED
    kbd.release(k)  # send keycode

while True:
    # check each button
    for button in buttons:
        i = buttons.index(button)
        if button.value is False:  # button is pressed?
            buttonspressed[i] = True  # save pressed button
            # was button not pressed last time?
            if buttonspressedlast[i] is False:
                print("Pressed #d" % i)
                pressbutton(i)
        else:
            buttonspressed[i] = False  # button was not pressed
            if buttonspressedlast[i] is True:  # was button pressed last time?
                print("Released #d" % i)
                releasebutton(i)

    #lightneopixels()  
    # save pressed buttons as pressed last
Upload the Code and Libraries to the KBRP2040

After downloading the Project Bundle, plug your KBRP2040 into the computer USB port. You should see a new flash drive appear in the computer's File Explorer or Finder (depending on your operating system) called CIRCUITPY. Unzip the folder and copy the following items to the KBRP2040's CIRCUITPY drive.

- lib folder
- code.py

Your KBRP2040 CIRCUITPY drive should look like this after copying the lib folder and code.py file.

![CIRCUITPY Drive Content]

3D Printing

Parts List
STL files for 3D printing are oriented to print "as-is" on FDM style machines. Parts are designed to 3D print without any support material. Original design source may be downloaded using the links below.
Slicing Parts

Slice with setting for PLA material.

The parts were sliced using CURA using the slice settings below.

- PLA filament 220°C extruder
- 0.2 layer height
- 10% gyroid infill
- 60mm/s print speed
- 60°C heated bed

Assemble

Solder wires

Measure and cut wires to fit the keyboard length.

Use a flush cutter to trim excess solder away from the sides and bottom of the board.

Press fit the KB2040 with the USB-C port facing outward.
Route wires

Pass wires through the opening on the side of the case.

Align the lid to the USB-C port opening on the case. Press fit the lid into the case to attach.
Mount Big Switch

Align the back of the Big Switch (LED side), to the USB-C port opening on the case. Carefully coil wires to side of the case. Press fit the Big Switch onto the case.

Attaching multiple case

Align the "T" connectors to attach them together. Pass wires from the main case to the next.
Connect Big Switch

Fit a quick connect wire on each of the pins on the Big Switch

Optional LED

A 10mm LED can be placed inside to illuminate the key when pressed.