Inhaltsverzeichnis

A Micro- Keypad out of a Senso- Game and a Raspi Pico

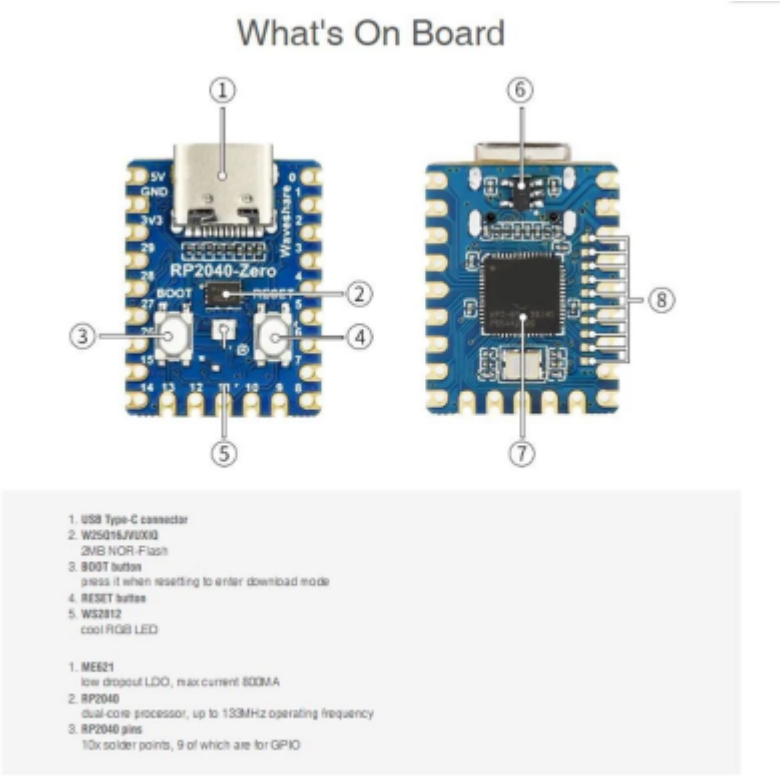
Inspired by the video from Lady Ada and the Adafruit DIY Pico Keypad tutorial, this little snipplet creates a small keypad with the most used finger breaking programmers key commands, which are

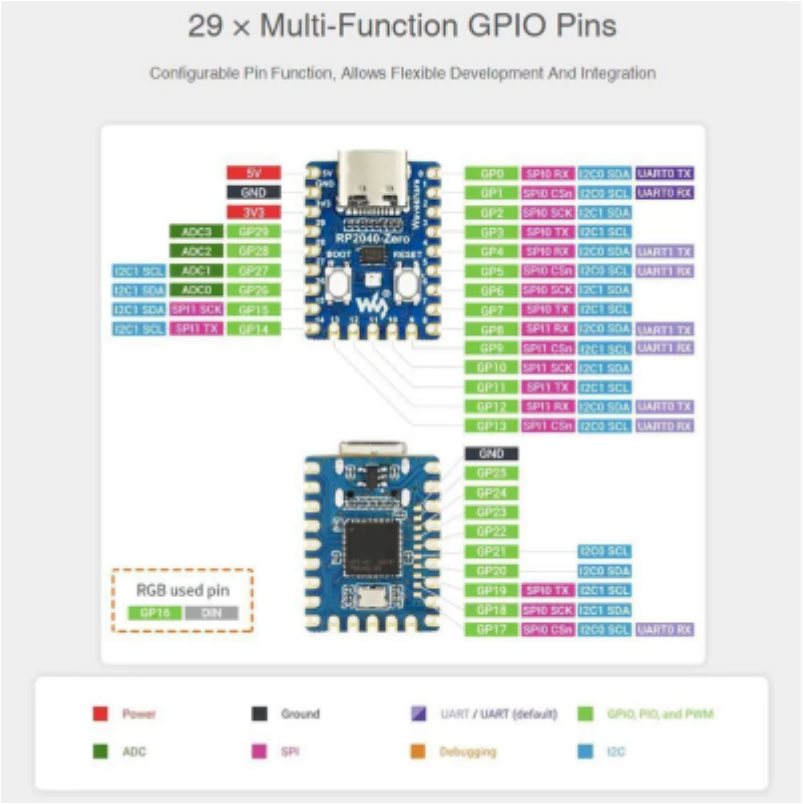
- CTRL + C (Copy)
- CTRL + V (Paste)
- CTRL + X (Cut)
- CTRL + Z (Undo)
- CTRL + L (Block Select in Visual Studio Code)

Key layout:

CTRL-X (RED)		CTRL-V (YELLOW)
	CTRL-L (Black)	
CTRL-Z (BLUE)		CTRL-C (GREEN)

As hardware base it took a chinese micro RP2040 board

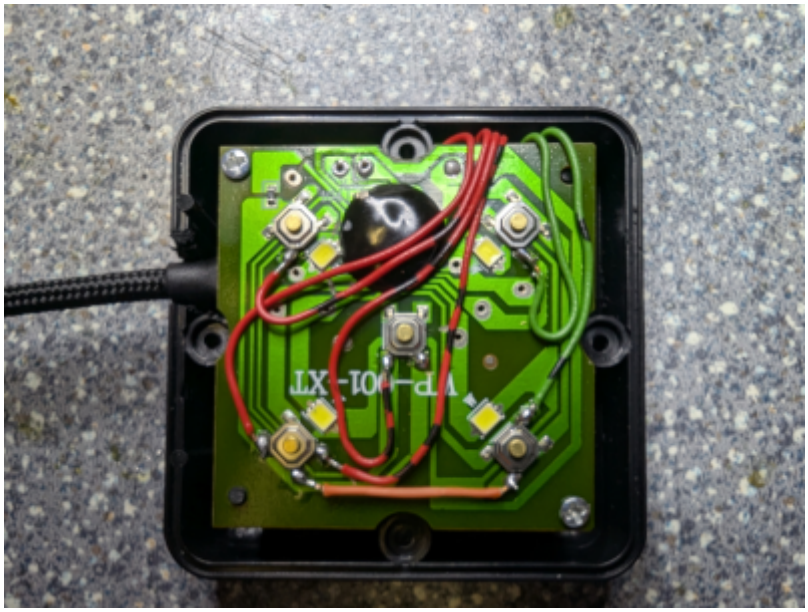




and this cheap Senso game (this is the original, unmodified color layout, not the final one!)

whick luckely provides a huge single sided PCB with a lot of space for modifications and storage space for the RP2040 (after removing the battery and loudspeaker holders)

The RP2040 is stored unterneath the pcb, on the pcb itself the connections to the switches are grinded away and new wires (switches and ground) are soldered between RP2040 and the pcb.



Install Circuit Python and the Adafruit HID library

Start with the RP2040 unplugged from USB. Hold down the B00TSEL button, and while continuing to hold it (don't let go!), plug the RP2040 into USB. Continue to hold the B00TSEL button until the RPI-RP2 drive appears!

If the drive does not appear, unplug your RP2040 and go through the above process again.

Download the latest UF2 file from the circuitpython homepage and install it by drag & drop onto your Pico.

A new drive is shown as `CIRCUITPI`.

Installing the Adafruit HID Library

Download the library bundle.

Copy the `adafruit_hid` folder from the bundle to the `lib` folder on your CIRCUITPY drive.

Before continuing make sure your board's lib folder has the adafruit_hid library folder copied over.

The Software

Store the following code as `code.py` on your `CIRCUITPI` drive. The code will start automatically and the keypad is now ready to use..

code.py

```
# stko: copied from
# SPDX-FileCopyrightText: 2021 John Park for Adafruit Industries
```

```
# SPDX-License-Identifier: MIT
# RaspberryPi Pico RP2040 Mechanical Keyboard

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

print("---Pico Joystick emulator Keyboard---")

led = DigitalInOut(board.LED)
led.direction = Direction.OUTPUT
led.value = True

kbd = Keyboard(usb_hid.devices)


pins = [
    board.GP28,  # black
    board.GP27,  # red
    board.GP26,  # green
    board.GP15,  # blue
    board.GP14,  # yellow
]

keymap = [
    Keycode.L,  #
    Keycode.X,
    Keycode.C,
    Keycode.Y,  # should be Keycode.Z, but to adjust it to german
keyboard settings...
    Keycode.V,
]

nr_of_keys = len(keymap)
switches = []
switch_state = []

for i in range(nr_of_keys):
    switch_state.append(0)
    switches.append(DigitalInOut(pins[i]))
    switches[i].direction = Direction.INPUT
    switches[i].pull = Pull.UP

while True:
    for button in range(nr_of_keys):
        if switch_state[button] == 0:
            if not switches[button].value:
                print(button, "on")
```

```
                try:
                    # kbd.press(keymap[button])
                    kbd.send(Keycode.CONTROL, keymap[button])
                except ValueError:  # deals w six key limit
                    print("e1")
                    pass
                switch_state[button] = 1

        if switch_state[button] == 1:
            if switches[button].value:
                print(button, "off")
                try:
                    pass  # kbd.release(keymap[button])

                except ValueError:
                    print("e1")
                    pass
                switch_state[button] = 0

    time.sleep(0.01)  # debounce
```