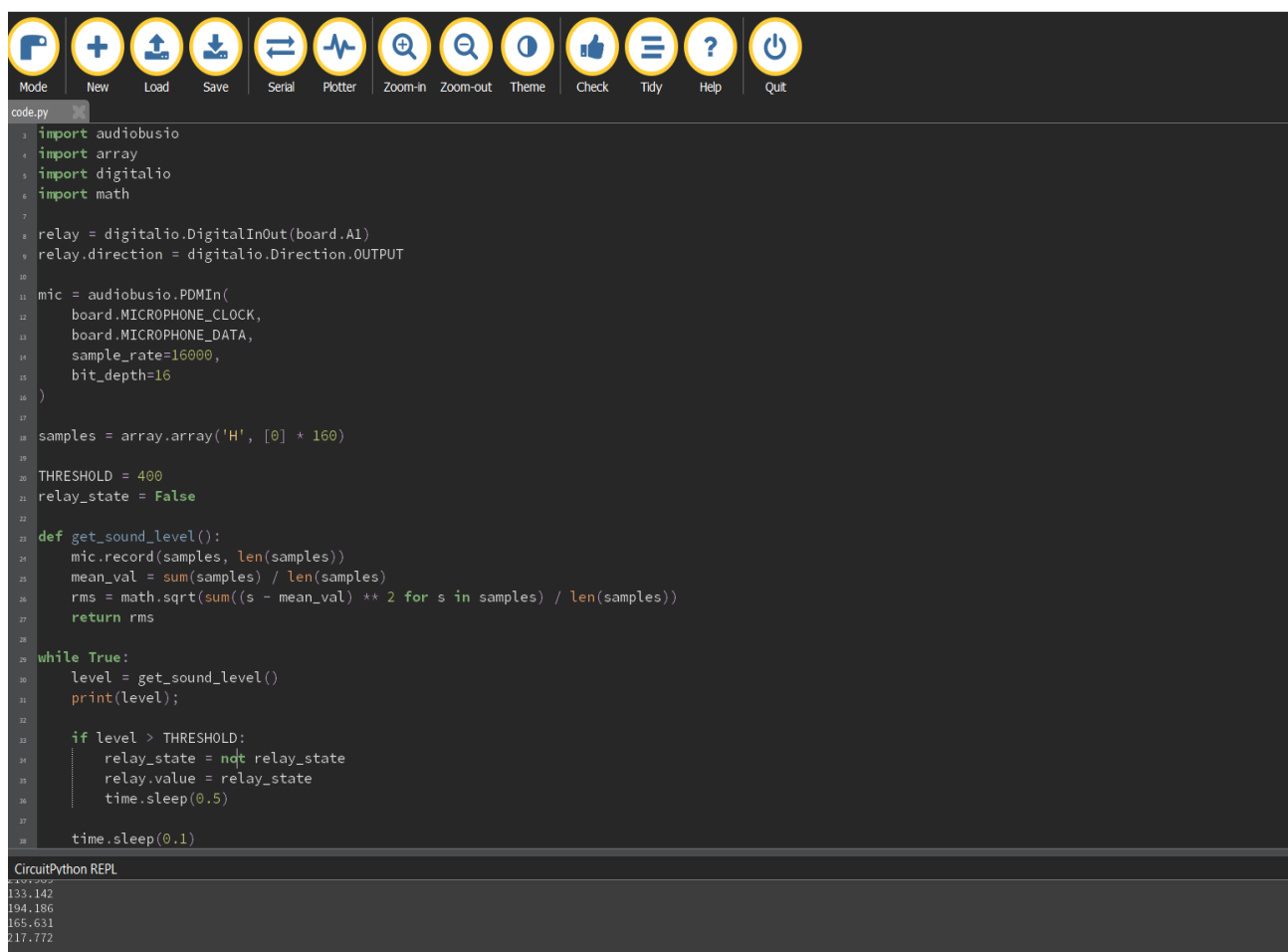


Clap-Controlled Relay – Simple Explanation

What This Program Does

This program turns an external light ON or OFF using a relay when a loud sound (like a clap) is detected by the CPX.

- The microphone keeps recording audio.
- It measures how loud the sound is using RMS.
- If sound level is greater than **400**, it treats it as a clap.
- If relay is OFF → turns ON
- If relay is ON → turns OFF
- Prevents multiple triggers from one clap.



```
code.py
1 import audiobusio
2 import array
3 import digitalio
4 import math
5
6 relay = digitalio.DigitalInOut(board.A1)
7 relay.direction = digitalio.Direction.OUTPUT
8
9 mic = audiobusio.PDMIn(
10     board.MICROPHONE_CLOCK,
11     board.MICROPHONE_DATA,
12     sample_rate=16000,
13     bit_depth=16
14 )
15
16 samples = array.array('H', [0] * 160)
17
18 THRESHOLD = 400
19 relay_state = False
20
21 def get_sound_level():
22     mic.record(samples, len(samples))
23     mean_val = sum(samples) / len(samples)
24     rms = math.sqrt(sum((s - mean_val) ** 2 for s in samples) / len(samples))
25     return rms
26
27 while True:
28     level = get_sound_level()
29     print(level)
30
31     if level > THRESHOLD:
32         relay_state = not relay_state
33         relay.value = relay_state
34         time.sleep(0.5)
35
36     time.sleep(0.1)
```

CircuitPython REPL

```
133.142
194.186
165.631
17.772
```

Step 1: Importing Required Libraries

Used to **read sound from the microphone**.

Used to **store multiple sound samples** in a list-like structure.

Used to **control digital pins** (like turning the relay ON/OFF).

Used for **calculations**, especially to find sound intensity (RMS).

Used to **add delays** (like `sleep`) to control timing.
Used to **identify hardware pins** (like A1, MIC pins).

Step 2: Setting Up the Relay

- Connects the relay to **pin A1** of the board.
- This pin will control the relay.
- Sets the pin as an **OUTPUT**.
- This allows the program to **send signals** (ON/OFF) to the relay.

This step is used to **initialize and control the relay module**

Step 3: Setting a Sound Limit (Threshold)

THRESHOLD = 400

THRESHOLD is a fixed value (limit).

- The program compares the **sound level** with this value.
- Sound level > **400** → Clap detected ✓
- Sound level < **400** → Ignore ✗

Step 4: Running the Program Continuously

while True:

- Creates an **infinite loop**.
- The program runs continuously (never stops).
- **level = get_sound_level()**
- Calls the function to **measure current sound level**.
- **print(level)**
- Displays the sound level in the console (for testing/debugging).

Step 5: Checking the Sound Level

- Captures sound from the microphone
- Stores it in the `samples` array
- Calculates the **average value of sound samples**
- **Measures actual sound intensity**
- Higher RMS → louder sound
- Lower RMS → softer sound
- Sends the sound level back to the main program

Step 6: Detecting a Clap

level → Current sound level (from microphone)

THRESHOLD → Predefined limit (e.g., 400)

The program compares both values:

If **sound level > threshold** → Clap detected ✓

If **sound level** $\leq$ **threshold** $\rightarrow$ No clap ✕

Step 7: Toggling the Relay

if level > THRESHOLD:

Checks if the sound is **loud enough (clap)**

relay_state = not relay_state

Changes the state:

- If OFF $\rightarrow$ becomes ON
- If ON $\rightarrow$ becomes OFF

relay.value = relay_state

Sends signal to relay:

- True $\rightarrow$ Relay ON
- False $\rightarrow$ Relay OFF

- **time.sleep(0.5)**
- Waits for 0.5 seconds
- Prevents one clap from triggering multiple times