HACKING MEANINGFUL CONNECTIONS WITH HUMANS BY TALKING TO (TOY) RODENTS

STEPHANIE NEMETH

TWITTER.COM/STEPHANIECODES · MASTODON.SOCIAL/@STEPH · STEPHANIE.LOL

STEPHANIE NEMETH

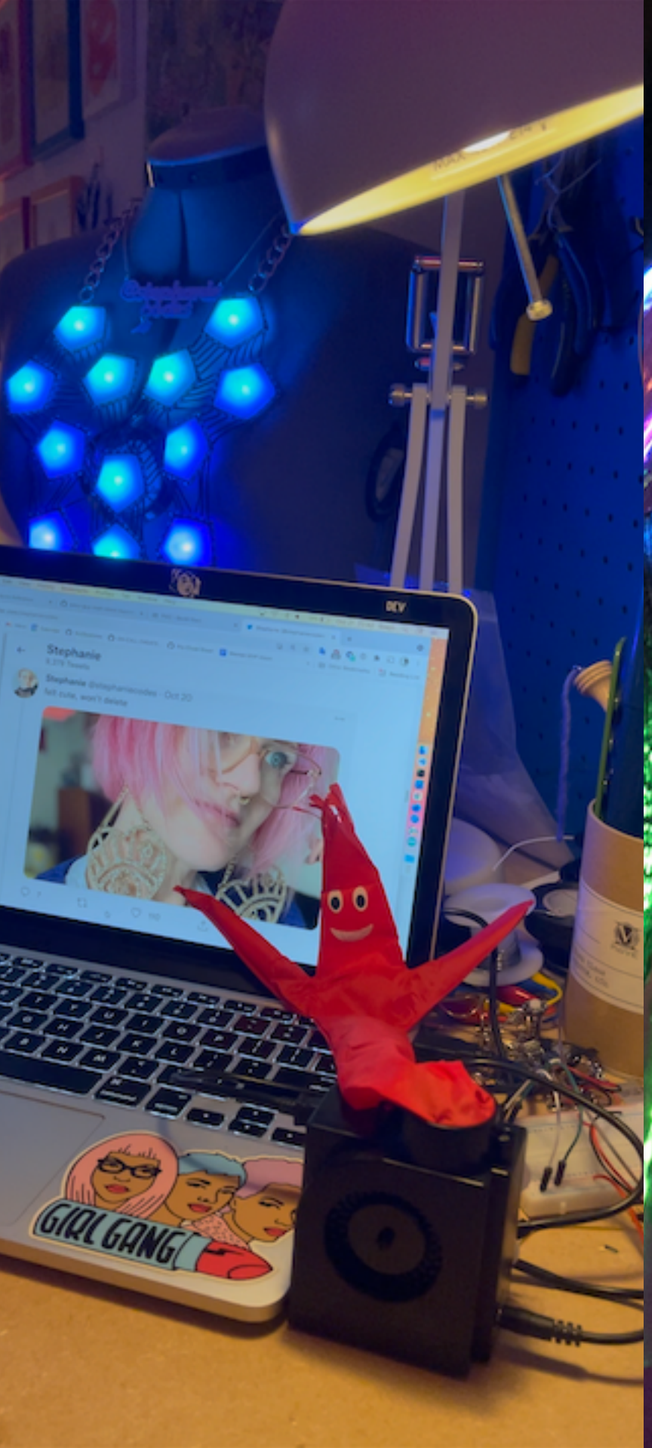
SOFTWARE ENGINEER @ GITHUB

SHE/HER

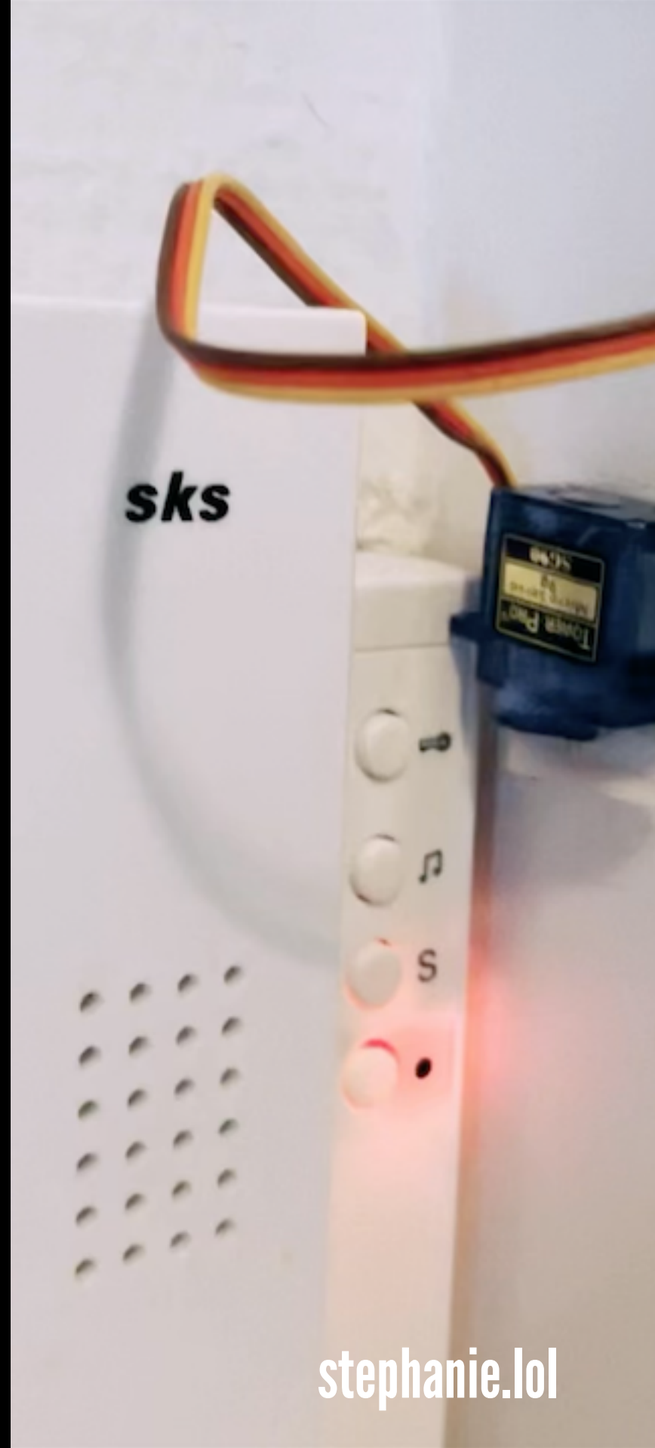
stephanie.lol

@steph@mastodon.social
@stephaniecodes

I ordered my first #raspberrypi today! And it has a rainbow case! (swoons) 😍😍🌈🌈 I'm gonna use it to build my #CS50 final project.

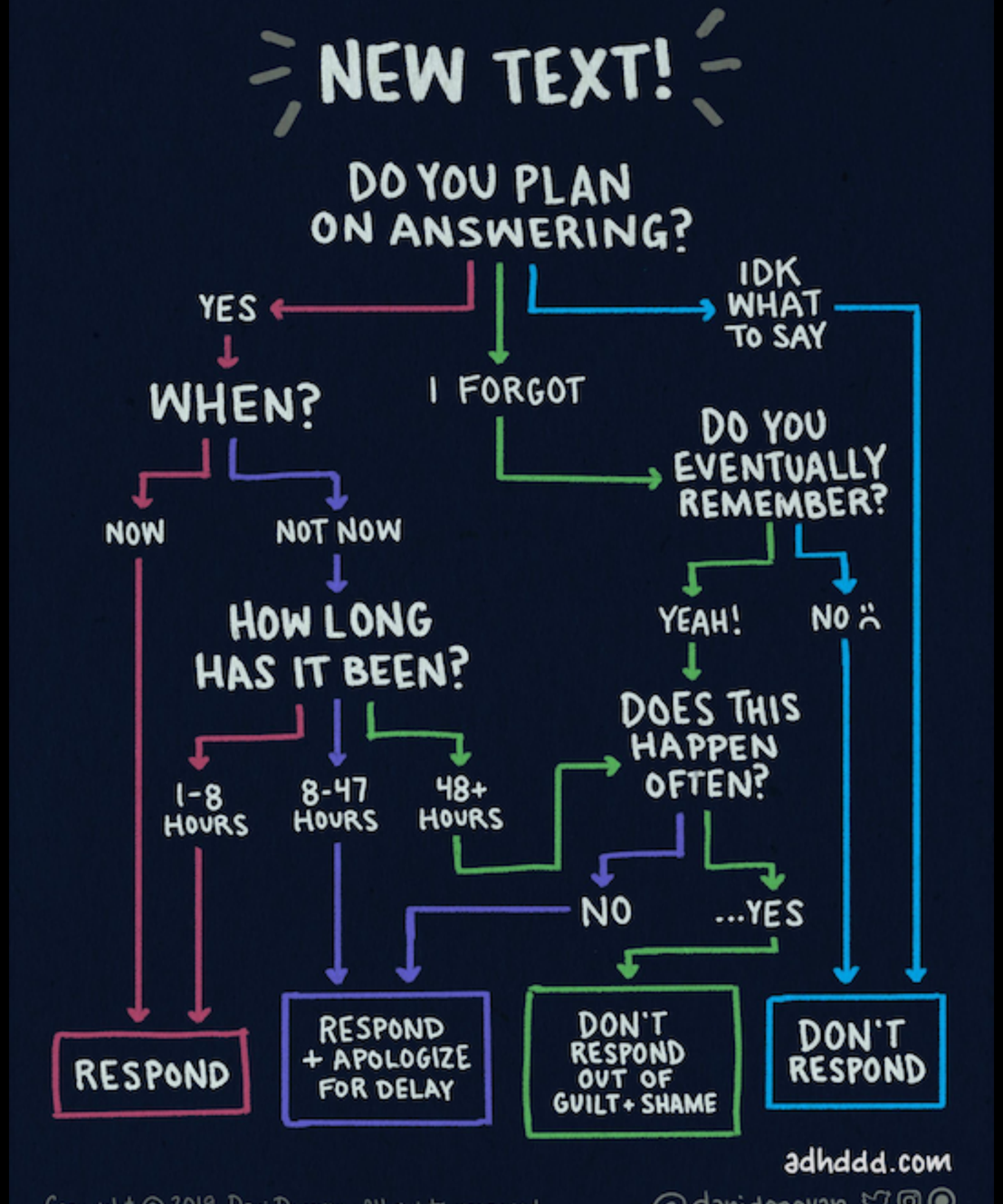

Stephanie
GIRL GANG


TINY ML IN THE WILD:
MACHINE LEARNING
PROJECTS FROM THE
COMMUNITY


sks
S
stephanie.lol

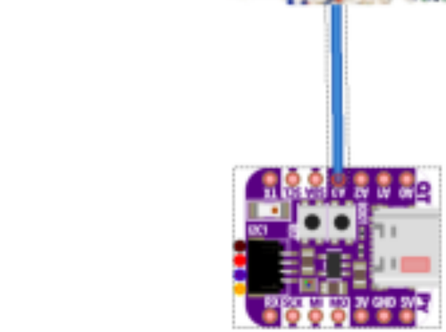
SOFT ELECTRONICS

stephanie.lol




GIANT POM POM THAT MAKES ART
My dear mushroom friend,
Despite life's ups and downs,
You can still cling to hope,
Hail the almighty cow.
stephanie.lol

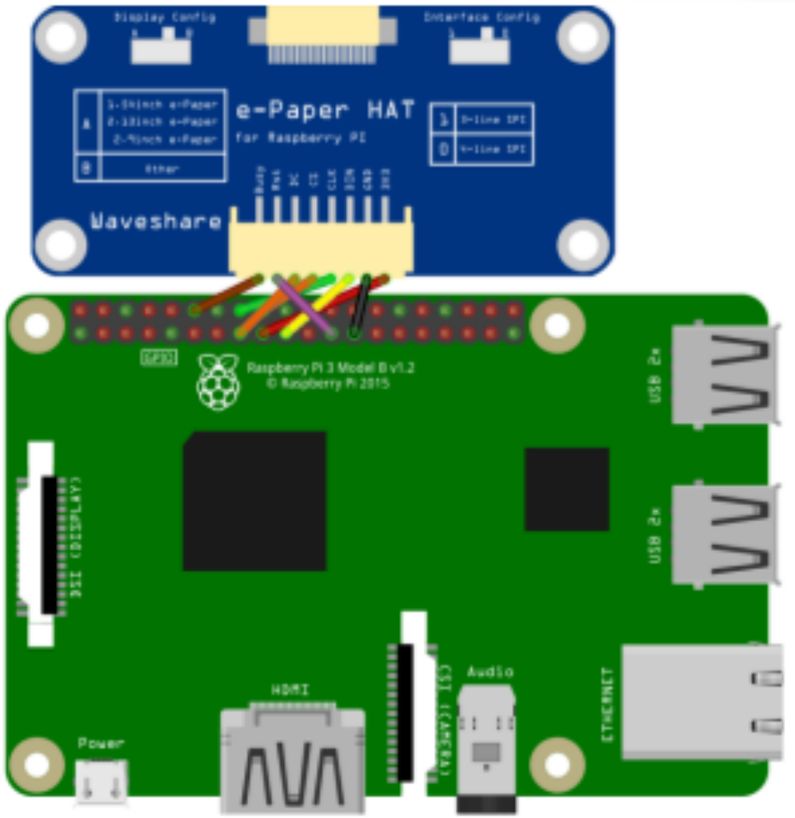

DEMO
stephanie.lol



Adafruit QT PY ESP32-S2

send message

to e-ink display



Raspberry Pi 3B

Displays new art

Makes request to ChatGPT API


ank
ung.
GRILLEON
stephanie.lol

MQTT

- lightweight publish/subscribe messaging protocol
- easy to setup in JS, circuitpython, arduino, etc
- consists of clients & a broker
- can build your own MQTT broker or use the cloud (eg HiveMQ)

Pom Pom - Circuitpython:

```
# Connect to wifi
network_connect()

# Connect to MQTT broker
client = MQTT.MQTT(
    broker=os.getenv('broker'),
    port=os.getenv('port'),
    username=os.getenv('user'),
    password=os.getenv('pw'),
    client_id=os.getenv("client_id"),
    socket_pool=pool,
    is_ssl=True,
    ssl_context=ssl.create_default_context(),
    )

client.connect()

if alarm.wake_alarm:
    client.publish("touch", "")

# Create an alarm that will trigger if pin is touched.
touch_alarm = alarm.touch.TouchAlarm(pin=board.A3)

# Disconnect from MQTT broker, go to  sleep again zzz
client.disconnect()
alarm.exit_and_deep_sleep_until_alarms(touch_alarm)
```

```
const client = mqtt.connect(process.env.MQTT_HOST, {
  username: process.env.MQTT_USER,
  password: process.env.MQTT_PASSWORD,
  clientId: process.env.CLIENT_ID,
  clean: false,
  reconnectPeriod: 1,
})

client.subscribe('touch')
client.subscribe('update')

async function handleTouch() {
  const generatedArt = await getCompletionFromOpenAI()
  client.publish('update', JSON.stringify(generatedArt))
}

client.on('message', function (topic, payload) {
  if (topic === 'update') {
    msg = JSON.parse(payload.toString())
    refreshDisplay()
  } else if (topic === 'touch') {
    handleTouch()
  }
})
```

```
let msg = ''

const configuration = new Configuration({ apiKey: process.env.OPENAI_API_KEY })
const openai = new OpenAIApi(configuration)

async function getCompletionFromOpenAI() {
  try {
    const completion = await openai.createChatCompletion({
      model: 'gpt-3.5-turbo',
      temperature: 1.2,
      messages: [
        {
          role: 'user',
          content: process.env.OPENAI_CONTENT,
        },
      ],
    })
    return completion.data.choices[0].message.content
  } catch (error) {
    if (error.response) {
      console.log(error.response.data)
    } else {
      console.log(error.message)
    }
  }
}
```

```
async function refreshDisplay() {
  const displayDevice = getDisplay()
  displayDevice.connect()

  const browserPage = await getPageRpi(
    displayDevice.width,
    displayDevice.height
  )

  const url = 'http://localhost:3000'
  const imgOfUrl = await browserPage.screenshot(url, { delay: 2000 })
  displayDevice.wake()
  await displayDevice.displayPng(imgOfUrl)
  displayDevice.sleep()
}

const __filename = fileURLToPath(import.meta.url)
const __dirname = path.dirname(__filename)
const dir = path.join(__dirname, '/ui')
const port = 3000

const app = express()

app.set('view engine', 'ejs', { async: true })

app.get('/', async (req, res) => {
  res.render(path.join(dir, 'index'), {
    chatbot_msg: msg,
  })
})
```

mastodon.social/@giantpompom

stephanie.lol

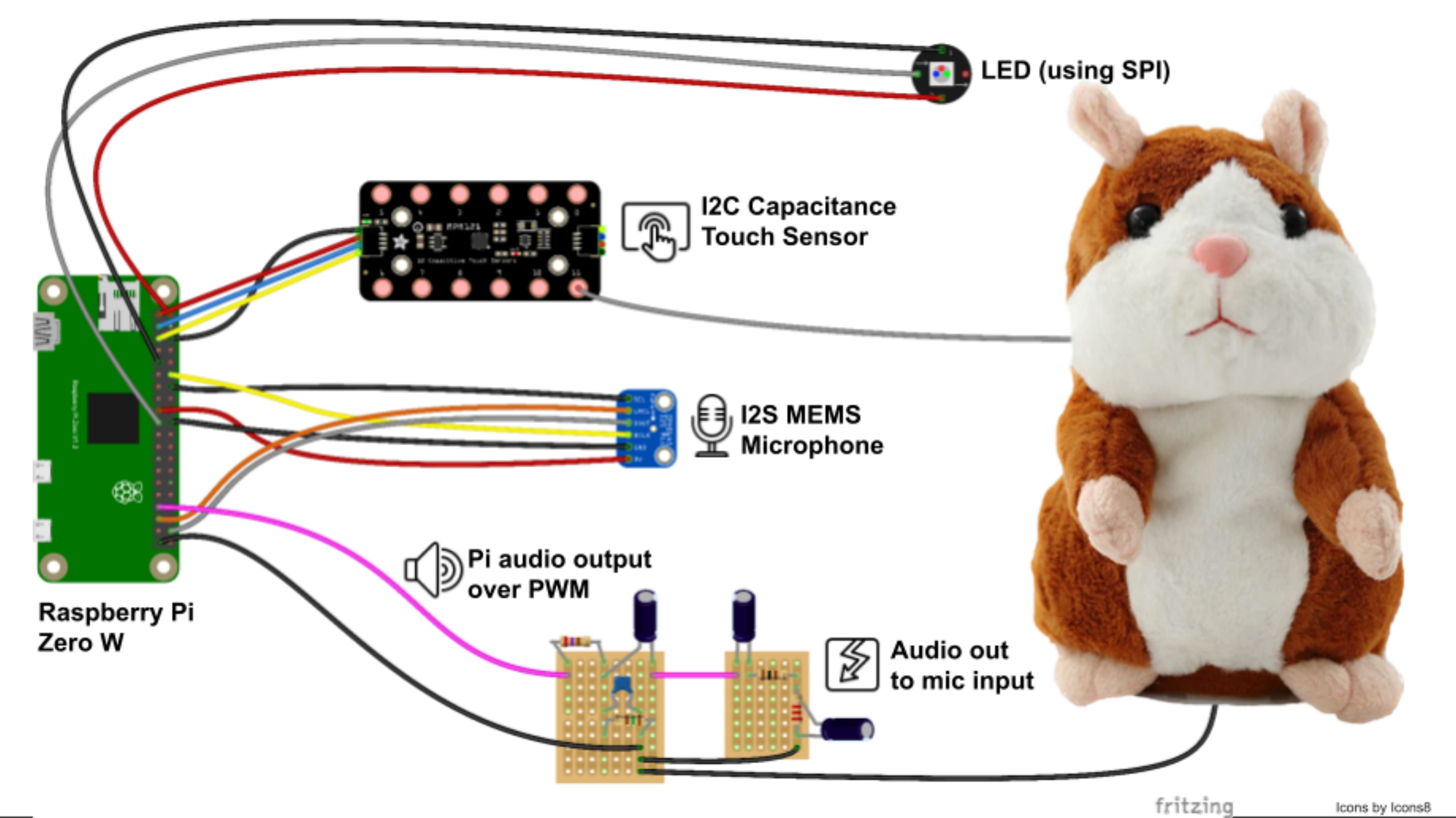

HAMSTER
WALKIE TALKIES
stephanie.lol

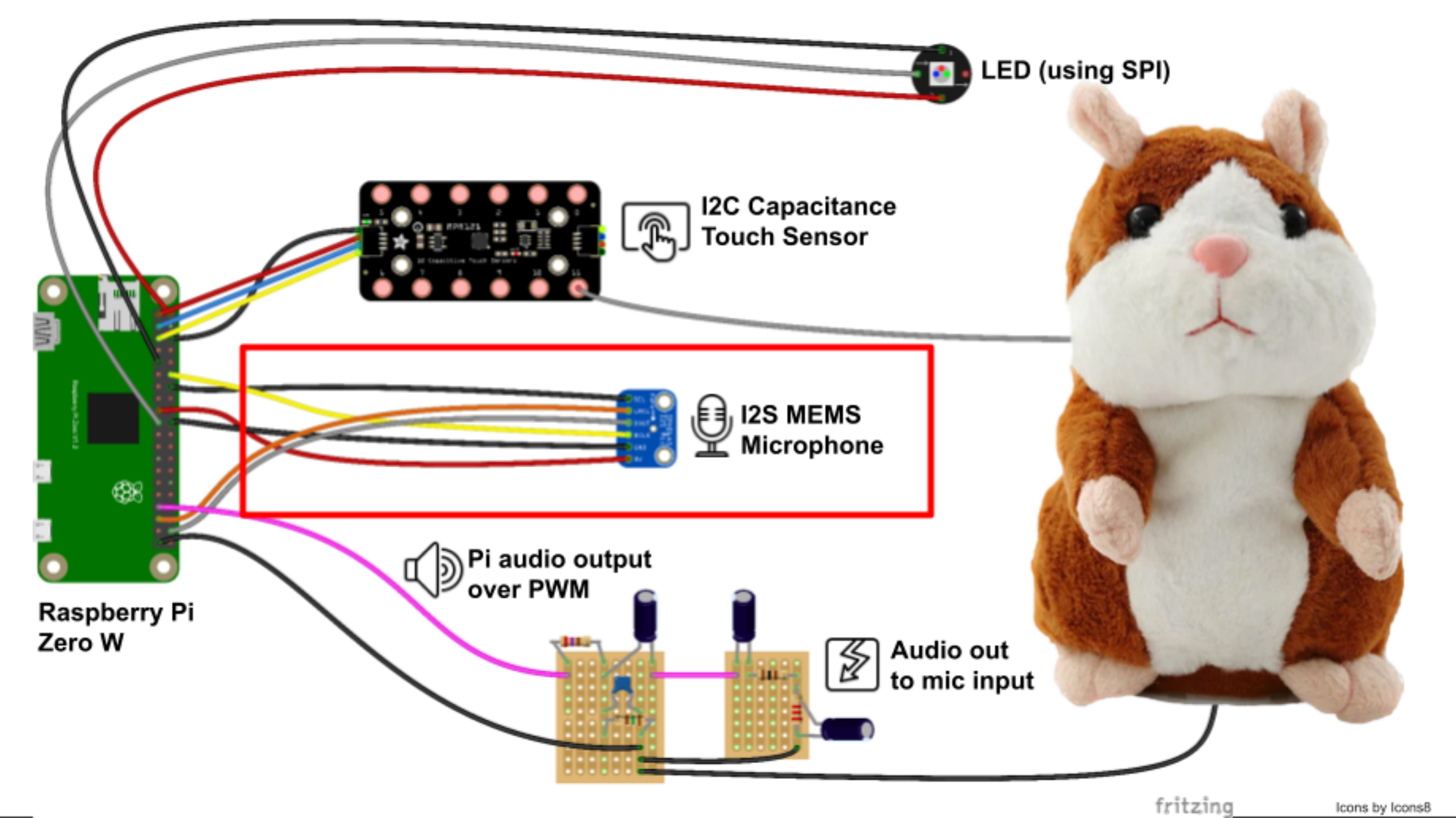
Teddy/Percy und sein Hamster

touch hamster to record audio

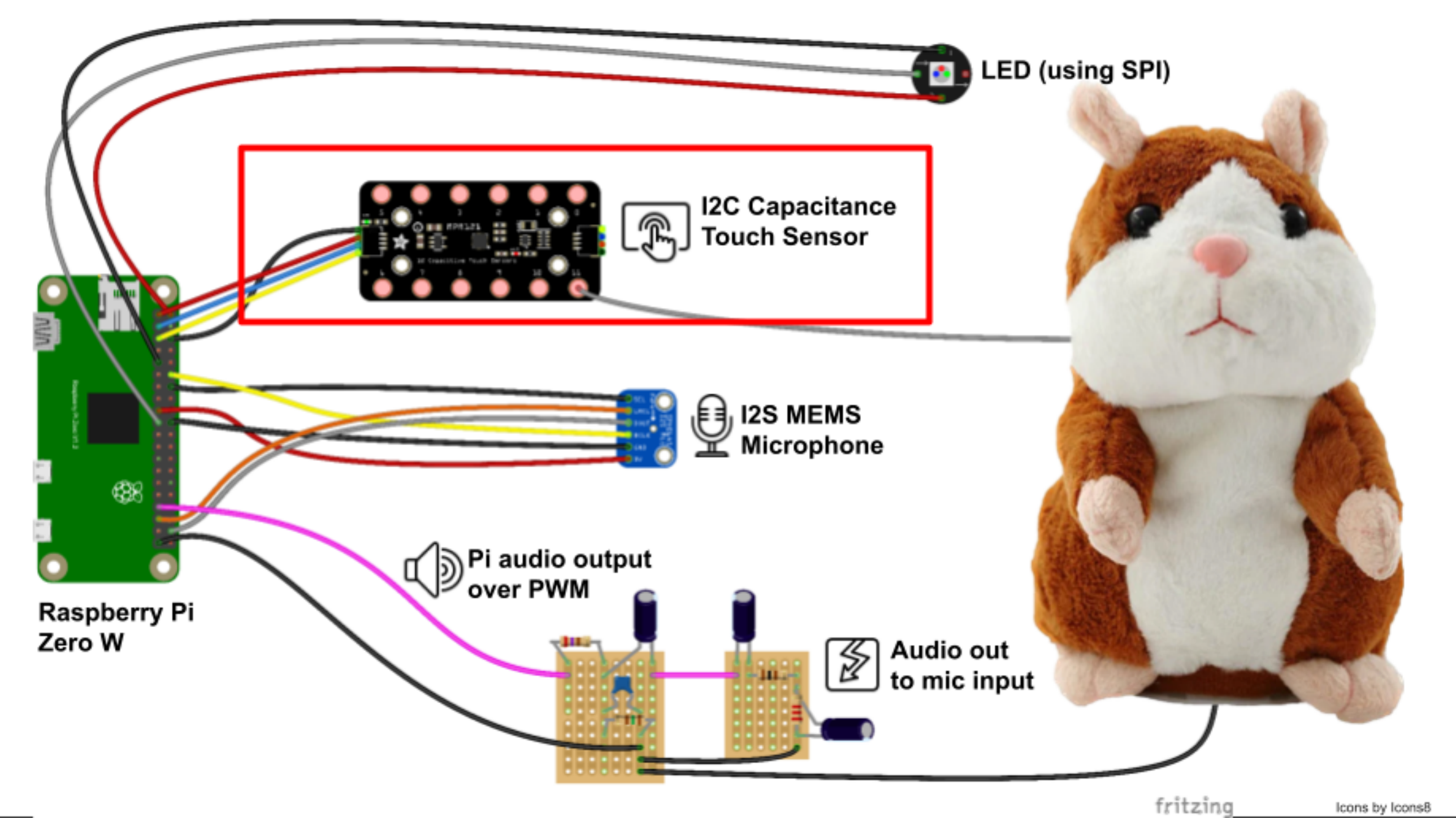
send audio

play audio


LED (using SPI)
I2C Capacitance
Touch Sensor
I2S MEMS
Microphone
Pi audio output
over PWM
Audio out
to mic input
Raspberry Pi
Zero W
fritzing
Icons by Icons8

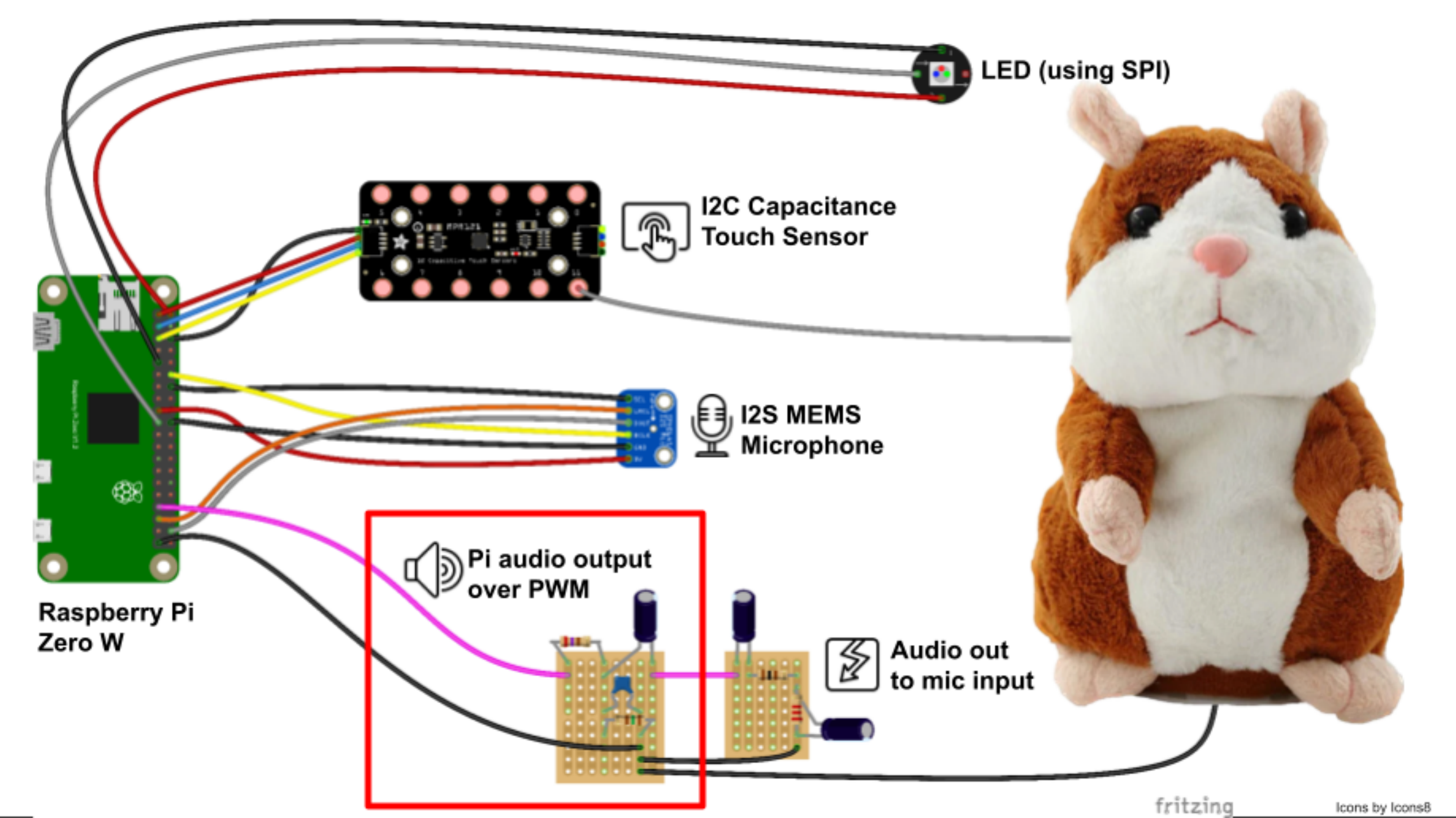

LED (using SPI)
I2C Capacitance
Touch Sensor
I2S MEMS
Microphone
Pi audio output
over PWM
Audio out
to mic input
Raspberry Pi
Zero W
fritzing
Icons by Icons8

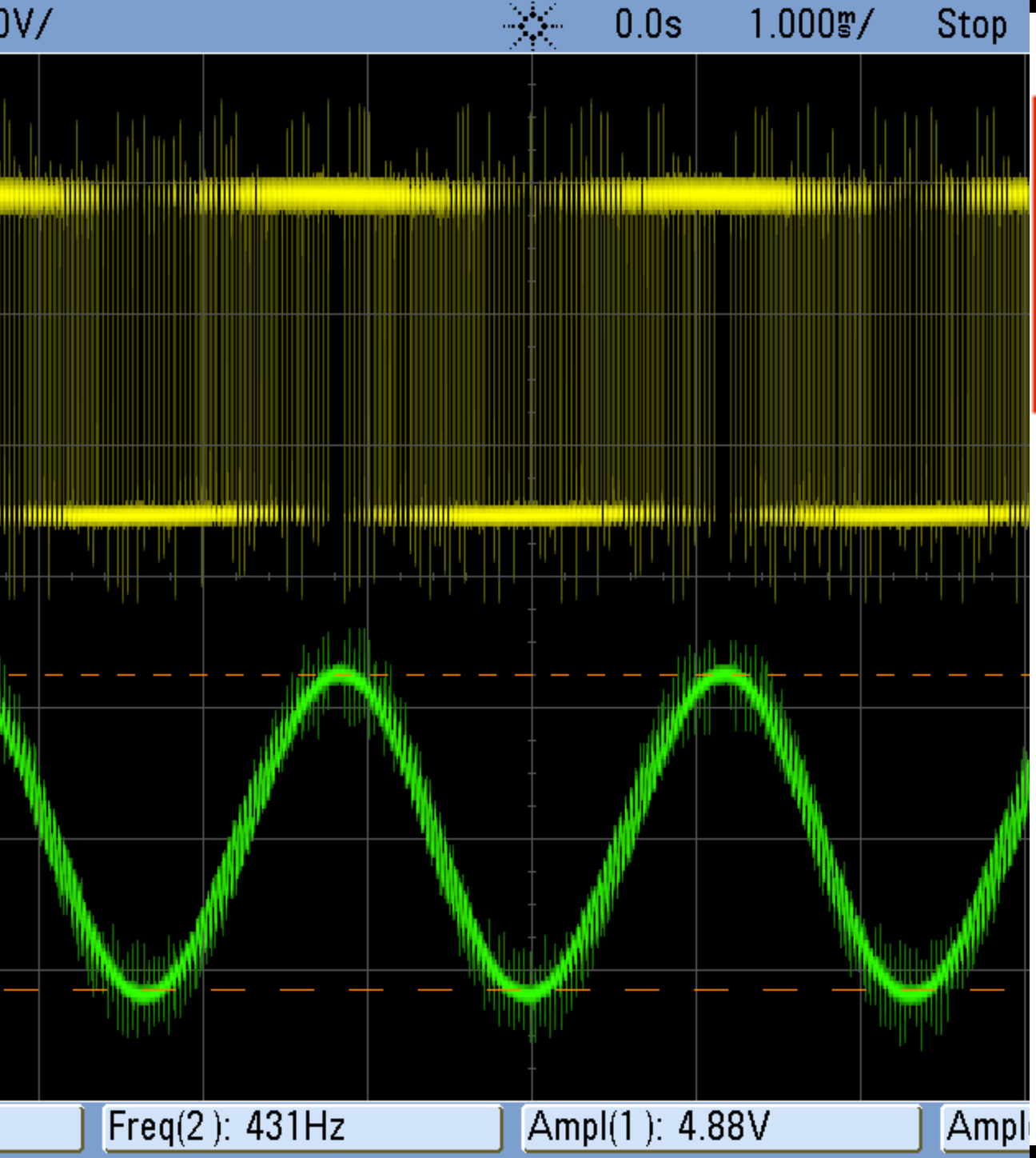
```
let childRecord;

const startRecording = () => {
  const voiceFile = `${new Date().getTime()}.wav`;

  childRecord = spawn("arecord", [
    "-D",
    "plughw:1",
    "-c1",
    "-r",
    "48000",
    "-f",
    "S32_LE",
    "-t",
    "wav",
    "-V",
    "mono",
    "-v",
    `${voiceFile}`,
  ]);

  childRecord.on("exit", function (code, sig) {
    if (code !== null && sig === null) {
      console.log("done recording");
    }
  });
  childRecord.stderr.on("data", function (data) {
    console.log("music record stderr :" + data);
  });
  childRecord.stdout.on("data", function (data) {
    console.log("music record stdout data: " + data);
  });
};

const stopRecording = () => {
  childRecord.kill("SIGTERM");
};
```


LED (using SPI)
I2C Capacitance
Touch Sensor
I2S MEMS
Microphone
Pi audio output
over PWM
Audio out
to mic input
Raspberry Pi
Zero W
fritzing
Icons by Icons8

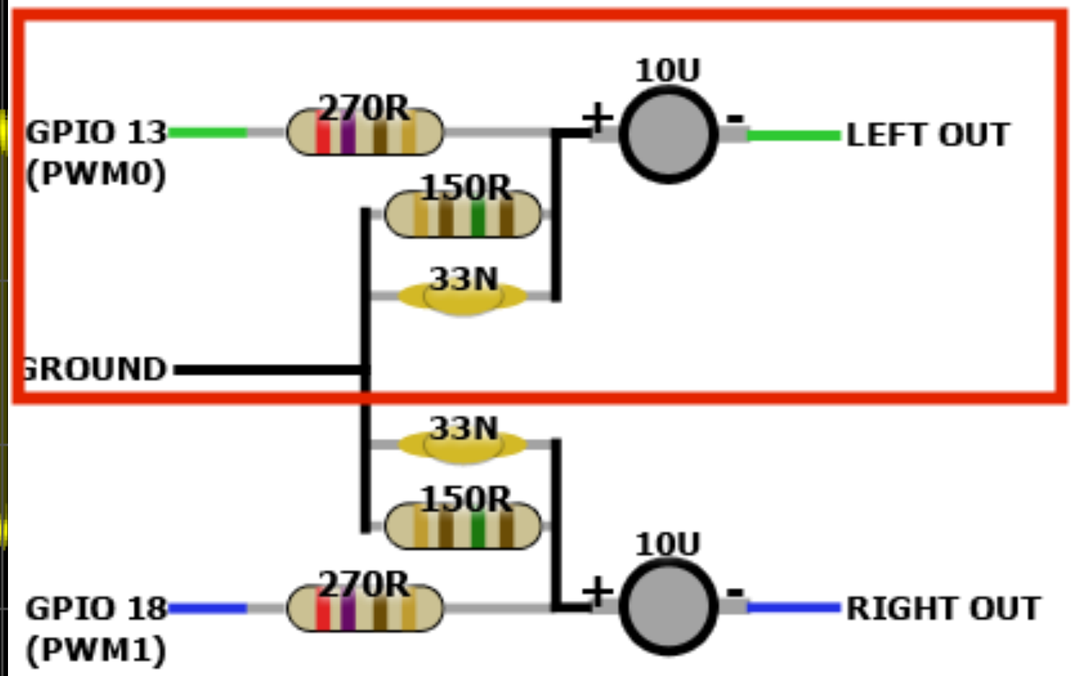
ADAFRUIT MPR121 CAPACITIVE TOUCH SENSOR

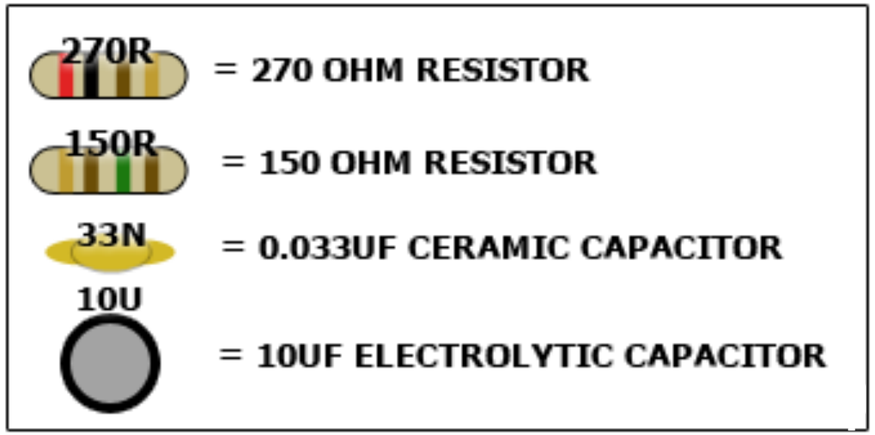
```
mpr121.on(11, (state) => {
  if (state) {
    startRecording();
  } else if (!state) {
    stopRecording();
  }
});
```

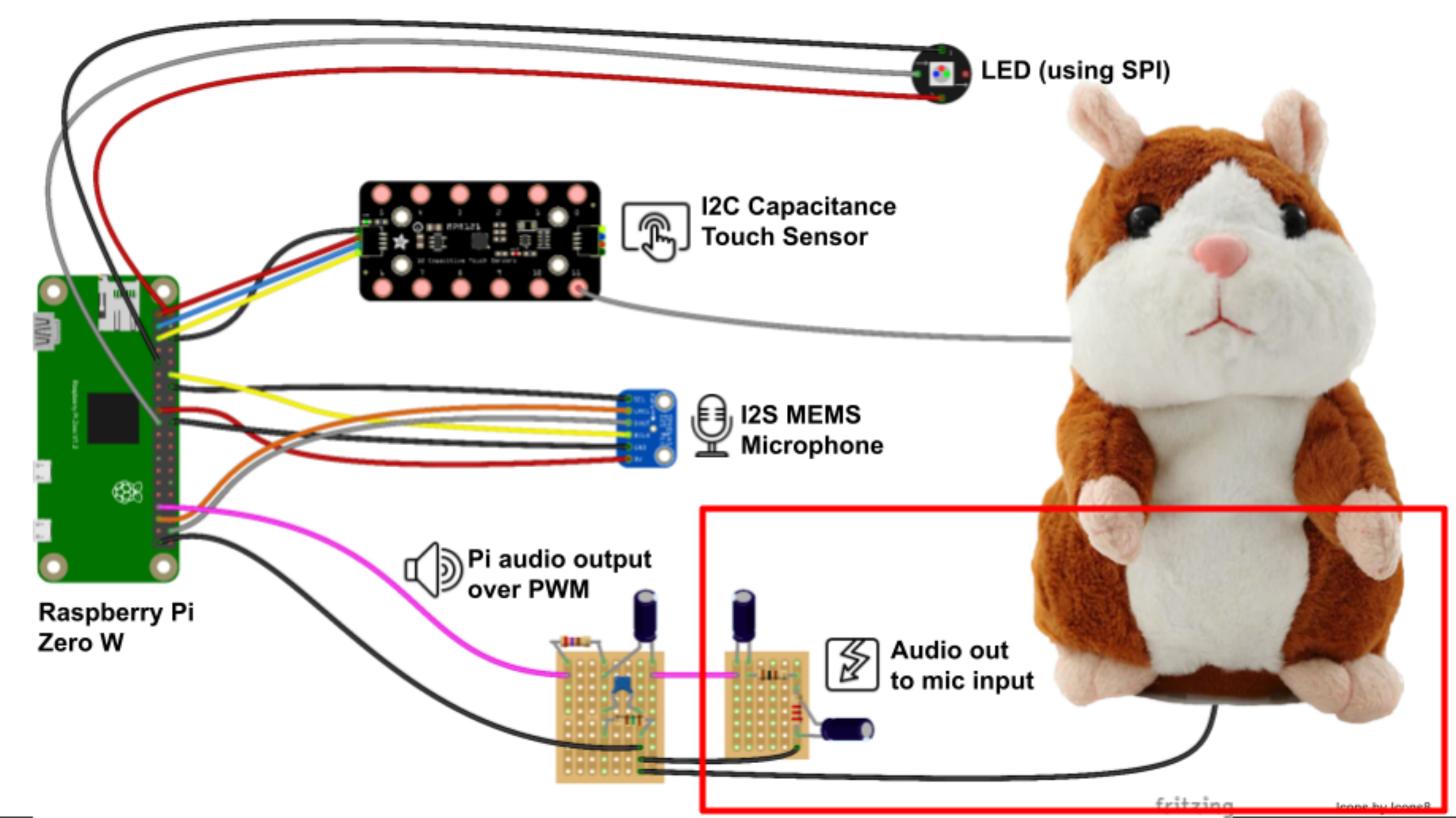

LED (using SPI)
I2C Capacitance
Touch Sensor
I2S MEMS
Microphone
Pi audio output
over PWM
Audio out
to mic input
Raspberry Pi
Zero W
fritzing
Icons by Icons8

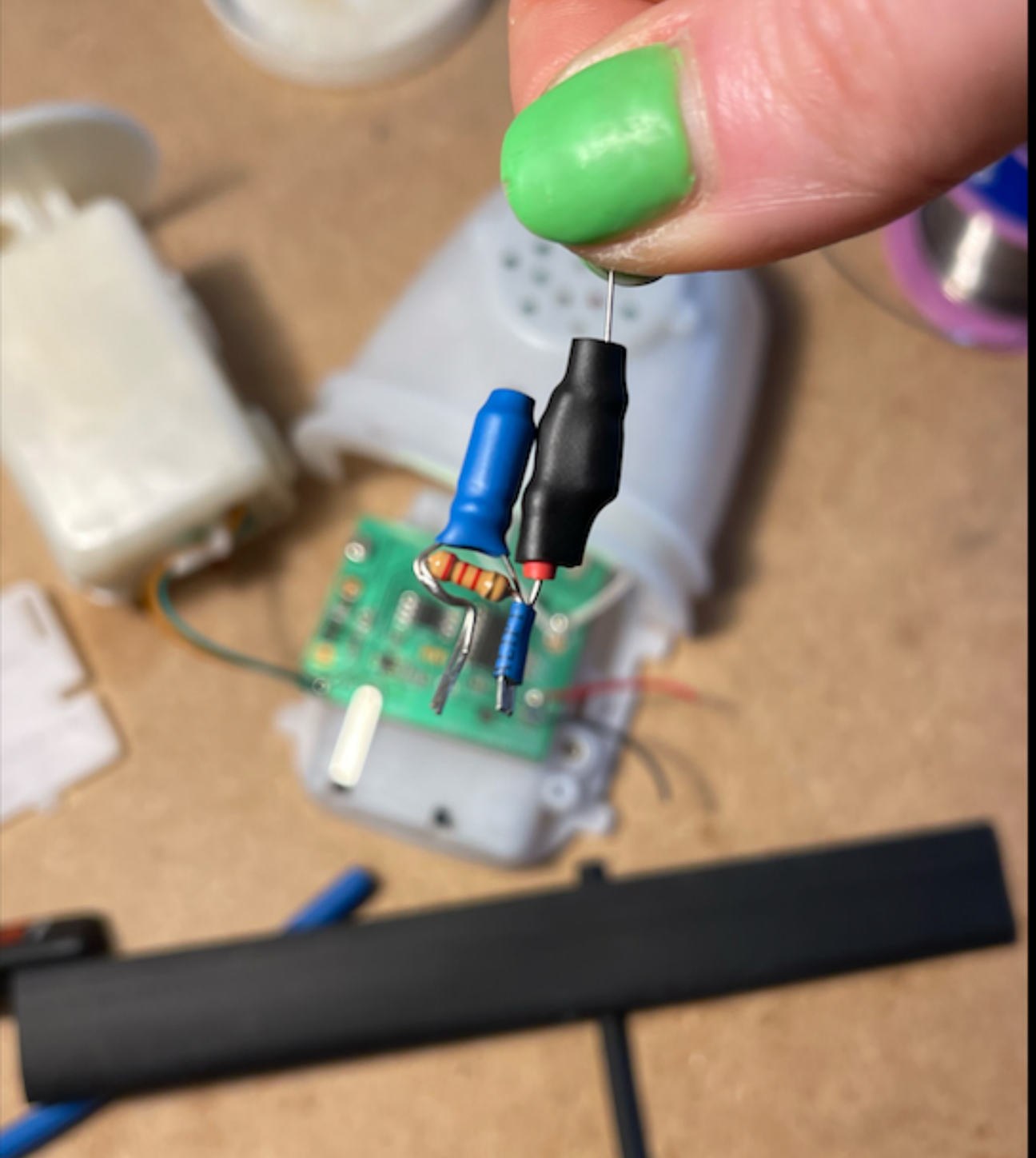


RASPBERRY PI ZERO PWM AUDIO SIMPLIFIED



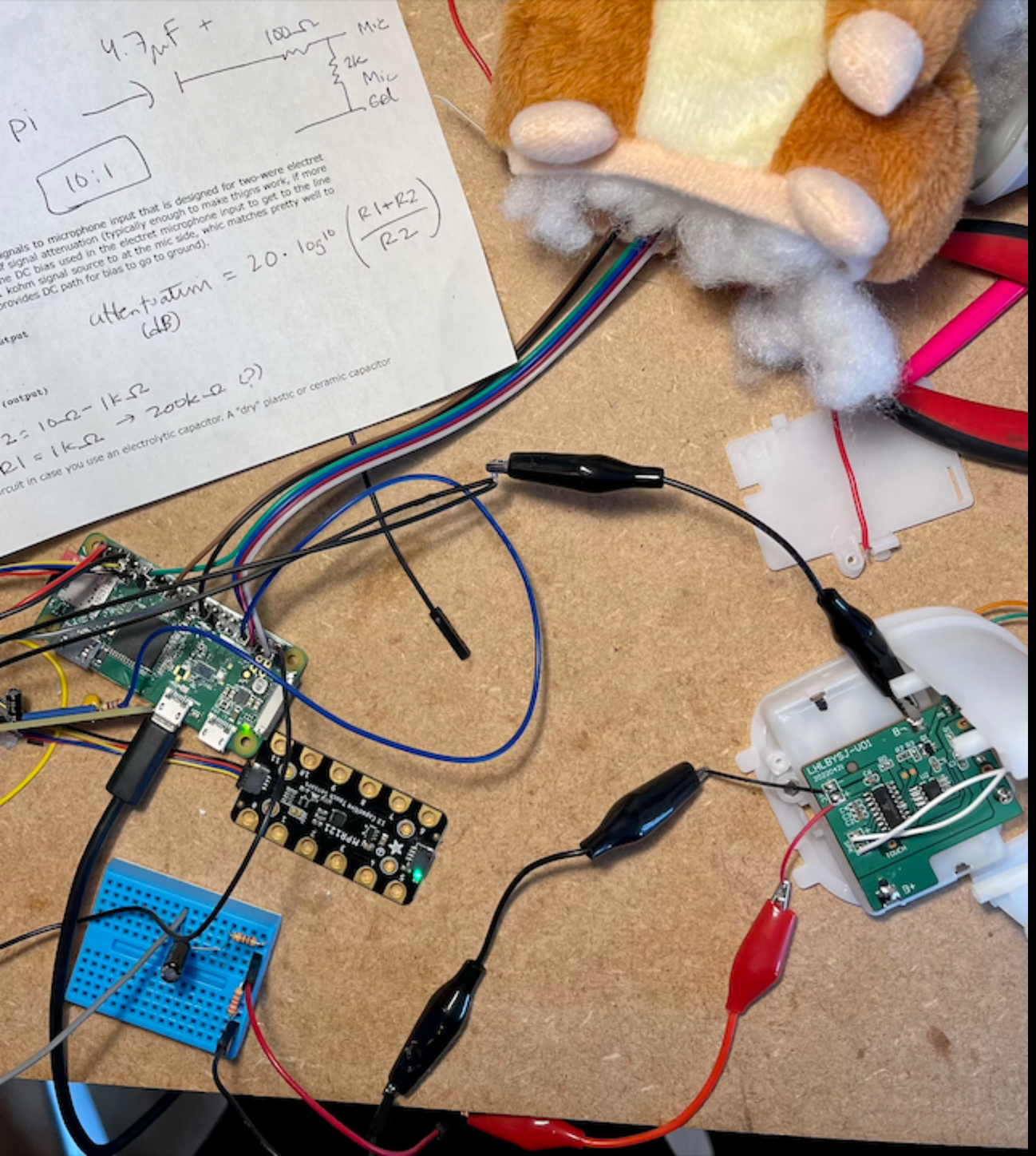



LED (using SPI)
I2C Capacitance
Touch Sensor
I2S MEMS
Microphone
Pi audio output
over PWM
Audio out
to mic input
Raspberry Pi
Zero W
fritzing



- Audio output from pi = line level (0.5V - 2V)
- Mic level = around 10 millivolts
- Create a voltage divider to reduce the voltage of the output to match the mic input
- Lots of trial & error

stephanie.lol




```
const playFile = (payload) => {
  const file = payload;
  childPlay = spawn("aplay", [`${file}`]);
  childPlay.on("exit", function (code, sig) {
    if (code !== null && sig === null) {
      console.log("done playing");
    }
  });
  childPlay.stderr.on("data", function (data) {
    console.log("music record stderr :" + data);
  });
};
```

stephanie.lol

Publishes topic: “audio”
message

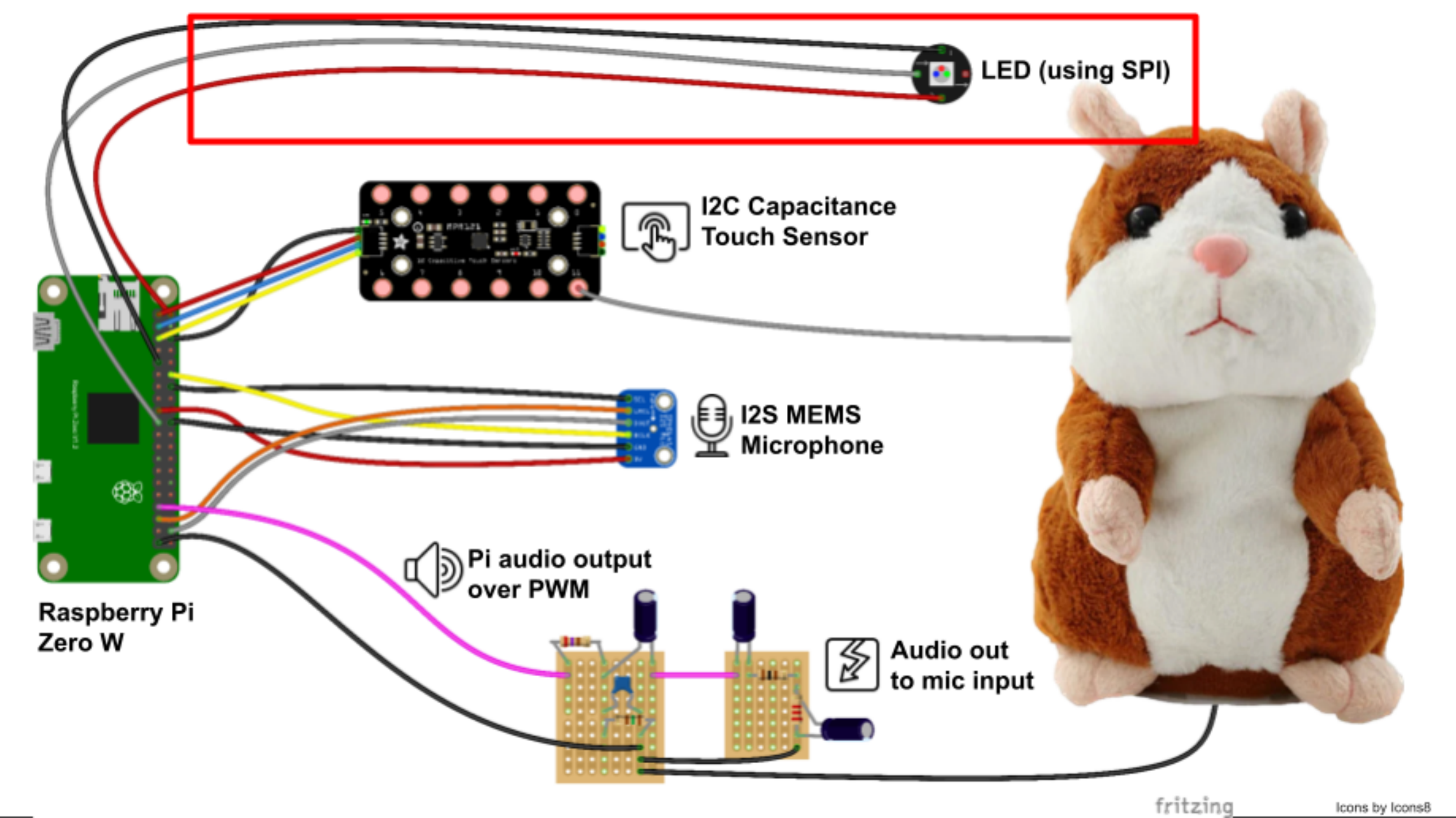
1. Touch hamster
2. Records audio
3. Send audio

Receives topic: “audio”
message

4. Receives audio
5. Plays audio

TRANSFORMING AUDIO FOR SENDING & RECEIVING MESSAGES

```
// recorded audio → other hamster
const sendFile = (voiceFile) => {
  const recordedBuffer = new WaveFile(fs.readFileSync(voiceFile)).toBuffer();
  client.publish(process.env.TOPIC, recordedBuffer);
};

// received audio → play
const createWavFile = async (bufferMsg) => {
  let receivedFile = `receivedFile-${new Date().getTime()}.wav`;
  await fs.promises.writeFile(receivedFile, bufferMsg);
  return receivedFile;
};
```


LED (using SPI)
I2C Capacitance
Touch Sensor
I2S MEMS
Microphone
Pi audio output
over PWM
Audio out
to mic input
Raspberry Pi
Zero W
fritzing
Icons by Icons8

stephanie.lol

THANKS! ✨

- github.com/traumverloren
- twitter.com/stephaniecodes
- mastodon.social/@steph
- bsky.social/stephaniecodes