Using LED Flash Timeout

Jul 2026

Introduction

*** Warning *** You can lock yourself out of your Pi Pico if you don't have a method to break the code. Please think about if you want/need to do this!!!

When we are using Thonny, we are loading our code in to an app and clicking a play button, when we want to stop the code (break out) we simply press the stop button in our Thonny App.

However, if the code runs as boot, you don't necessarily have this option, so we need to build in a fail safe. There are two that we can use (there are more I am sure) and they are.

- 1. Build in a delay at the start, where you can connect to Thonny and click the stop button.
- 2. Add a button to a GPIO pin, and when that button is pressed, stop the code.

We are going to look at the first option in this example. Please not, pressing the BOOTSEL button will not stop the code.

Please note, this behaviour changes depending on the age of your Pico, the library files you are using and the version of Thonny. With the latest versions of code, it seems that this is less of an issue. It is still a good option to have some built in safeguards though.

What we are going to do is to flash the onboard LED for 20 seconds before starting the main code. This gives you 20 seconds to connect the Pico to your computer, run Thonny and hit the stop button.

We have to use two different sets of code unfortunately, as the onboard LED on the Pico and Pico W/W2 use a different connection method.

- On the original Raspberry Pi Pico and Pico 2, the onboard LED is wired directly to GPIO25 on the RP2040.
- On the Pico W and Pico 2W the LED is controlled through the Wi-Fi chip, using a pin called WL_GPIO0.

So we will need a slightly different set of code to flash the LED.

Raspberry Pi Pico 1 / Pico 2

Below is the (breakout) code for the Pico1 and Pico2

| download

```
#######################################################
# visual Thonny breakout.

#use this declaration for Pico1 & Pico2

# Set Pin25 as output (Pin 25 is where the onboard LED is connected)
LED = Pin(25, Pin.OUT)

# Print a message in Thonny to remind user to break out of code now.
print('booting... Press ctrl-c to stop')

# loop 20 times
for i in range (20):

# turn on LED, wait half a second.
```

```
#for Pico1 & Pico2
    LED.on()
    time.sleep(0.5)

# turn off LED, wait half a second.
    LED.off()
    time.sleep(0.5)
#####################################################
```

What we eed to do now is to place this code, inside of our main program that we want to run on boot. In this example I am using the traffic light example, so when I power on the Pico, the traffic light sequence will start (after the initial 20 second delay).

Below, you can see where we have inserted this code block in to our Traffic Light code.

| download

```
#Traffic light LEDs using an array

from machine import Pin
import time

# Define GPIO pins for the LEDs
RED = Pin(2, Pin.OUT)
AMBER = Pin(6, Pin.OUT)
GREEN = Pin(10, Pin.OUT)

# Put them in an array
lights = [RED, AMBER, GREEN]

#####################################################
# visual Thonny breakout.

#use this declaration for Pico1 & Pico2

# Set Pin25 as output (Pin 25 is where the onboard LED is connected)
LED = Pin(25, Pin.OUT)

# Print a message in Thonny to remind user to break out of code now.
print('booting... Press ctrl-c to stop')

# loop 20 times
for i in range (20):

# turn on LED, wait half a second.
#for Pico1 & Pico2
    LED.on()
    time.sleep(0.5)

# turn off LED, wait half a second.
    LED.off()
    time.sleep(0.5)
#####################################################

# Helper function to turn all lights off
def all_off():
    for light in lights:
        light.off()

while True:
    # Red
    all_off()
    RED.on()
    time.sleep(3)

    # Red + Amber
    AMBER.on()
    time.sleep(1)

    # Green
```

```
    all_off()
    GREEN.on()
    time.sleep(5)

    # Amber
    all_off()
    AMBER.on()
    time.sleep(1)
```

Raspberry Pi Pico W and Pico 2W

For the wireless based Pico, we need to use slightly different code, as the LED is not on Pin25, it is accessed via a pin in the Wifi chip.

| download

```
#######################################################
# visual Thonny breakout.

#use this declaration for Pico1W & Pico2W

#use this declaration for Pico 1W and Pico 2W
OBLED = Pin('LEDW',Pin.OUT)

# Print a message in Thonny to remind user to break out of code now.
print('booting... Press ctrl-c to stop')

# loop 20 times
for i in range (20):


#for Pico1W and Pico2W use this
# turn on LED, wait half a second.
    OBLED.on()
    time.sleep(0.5)

# turn off LED, wait half a second.
    OBLED.off()
    time.sleep(0.5)
#######################################################
```

Once again, let us look at this code block embedded in our original Traffic Light code.

| download

```
#Traffic light LEDs using an array

from machine import Pin
import time

# Define GPIO pins for the LEDs
RED = Pin(2, Pin.OUT)
AMBER = Pin(6, Pin.OUT)
GREEN = Pin(10, Pin.OUT)

# Put them in an array
lights = [RED, AMBER, GREEN]


#######################################################
# visual Thonny breakout.

#use this declaration for Pico1W & Pico2W
```

```
#use this declaration for Pico 1W and Pico 2W
OBLED = Pin('LEDW',Pin.OUT)

# Print a message in Thonny to remind user to break out of code now.
print('booting... Press ctrl-c to stop')

# loop 20 times
for i in range (20):


#for Pico1W and Pico2W use this
# turn on LED, wait half a second.
    OBLED.on()
    time.sleep(0.5)

# turn off LED, wait half a second.
    OBLED.off()
    time.sleep(0.5)
#####################################################



# Helper function to turn all lights off
def all_off():
    for light in lights:
        light.off()

while True:
    # Red
    all_off()
    RED.on()
    time.sleep(3)

    # Red + Amber
    AMBER.on()
    time.sleep(1)

    # Green
    all_off()
    GREEN.on()
    time.sleep(3)

    # Amber
    all_off()
    AMBER.on()
    time.sleep(1)
```

In Action

From Thonny, I am going to start our traffic light code with the new breakout code block.