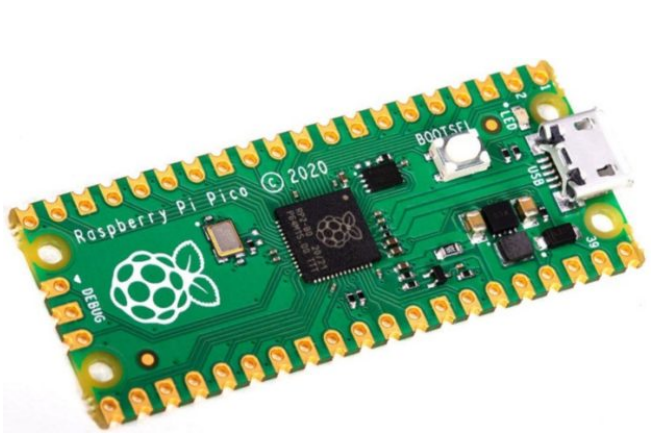
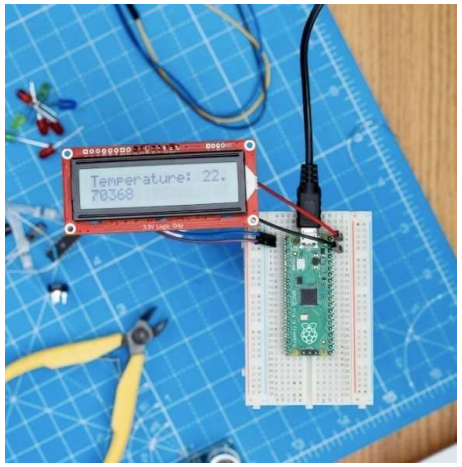
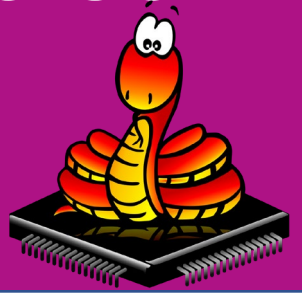


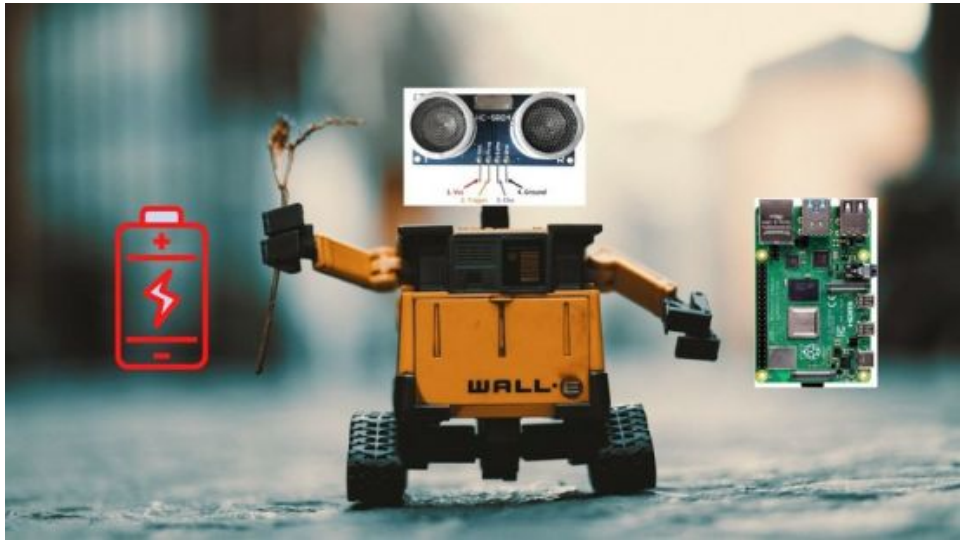


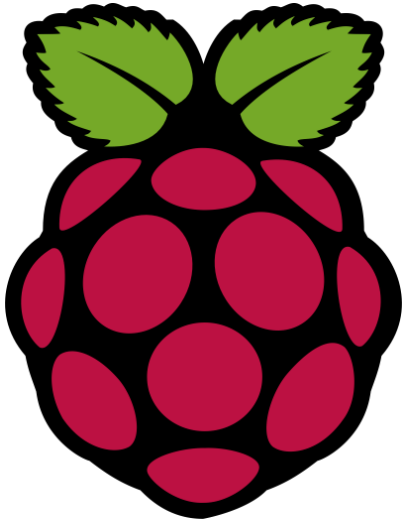
INTERNET OF THINGS (IOT) PROJECTS USING PYTHON

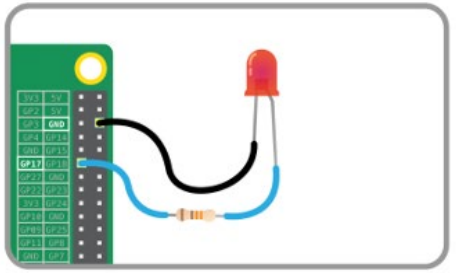
(CSE 4110)

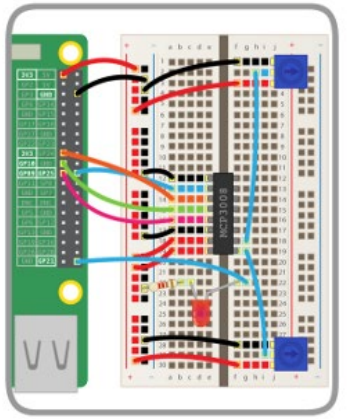
(LECTURE – 3)











Course Outcomes

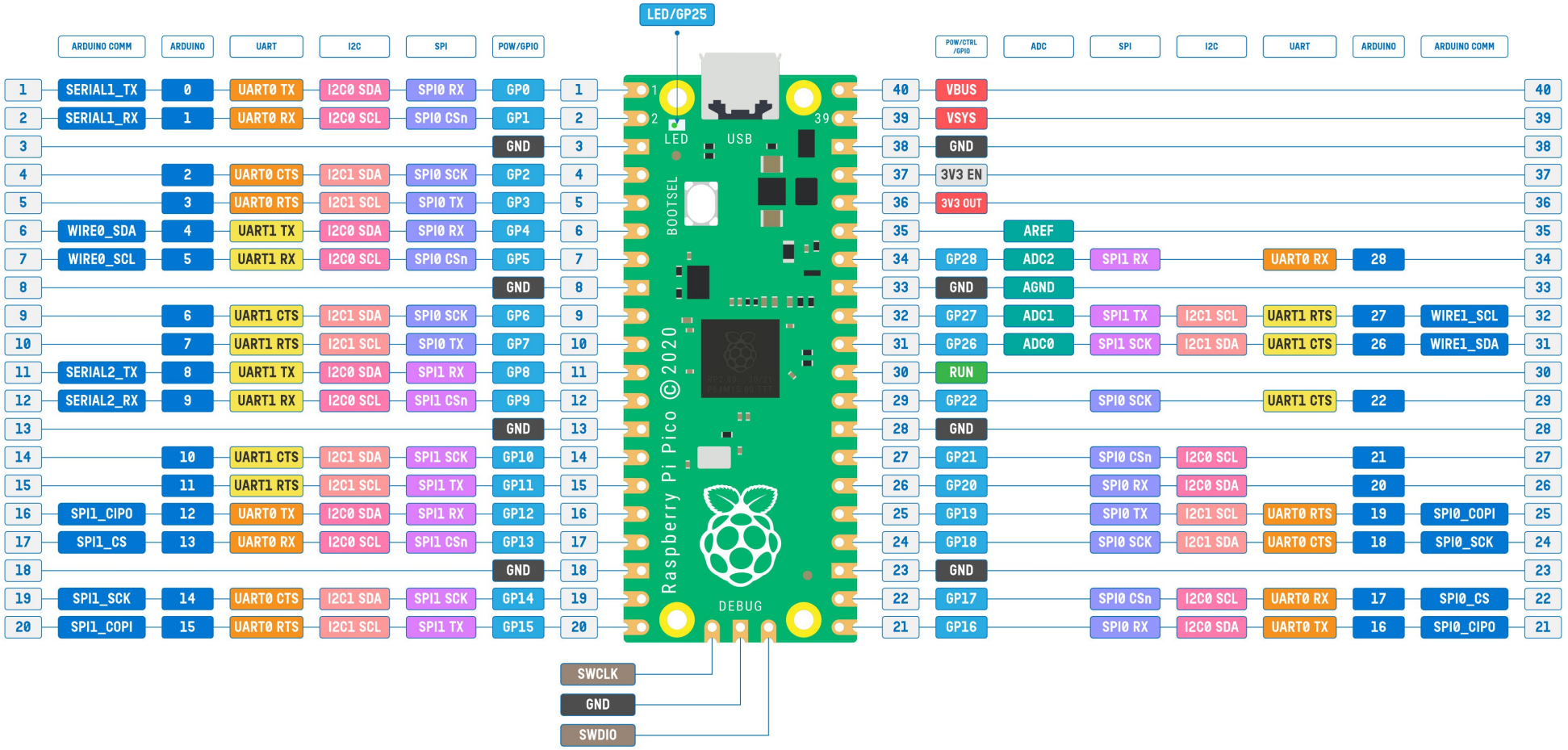
Course Outcomes	
CO1	Understand general concepts of Internet of Things (IoT), the working of Raspberry Pi and its features.
CO2	Recognize various components, sensors, actuators, devices and their applications.
CO3	Analyze various python programs to interface with sensors, actuators, LED's, cloud and camera using Raspberry pi.
CO4	Measure physical parameters using sensors.
CO5	Demonstrate the ability to transmit data wirelessly between different devices to build simple IoT systems using Raspberry Pi.
CO6	Create IoT devices and systems through a variety of interfaces, including web apps and mobile apps.

What Students will learn?

Students will be developing simple hardware projects/experiments with the Raspberry Pi Pico, using the Thonny text editor. They also learn how a Raspberry Pi Pico can be used to control and receive input from a variety of electronic components to engage in some creative physical computing experiments/projects.

- ✓ Implementation of Flashing LED using a Timer using Raspberry Pi Pico.
- ✓ Implementation of Alternately Flashing two different colour LEDs using Raspberry Pi Pico.
- ✓ Implementation of four LEDs display a pattern of rotating left using Raspberry Pi Pico

Raspberry Pi Pico – Full Pinout



*Raspberry Pi and the Raspberry Pi logo are trademarks of Raspberry Pi Ltd.

Raspberry Pi Pico vector image is originally designed by Raspberry Pi. Please visit **raspberrypi.com** for more info.

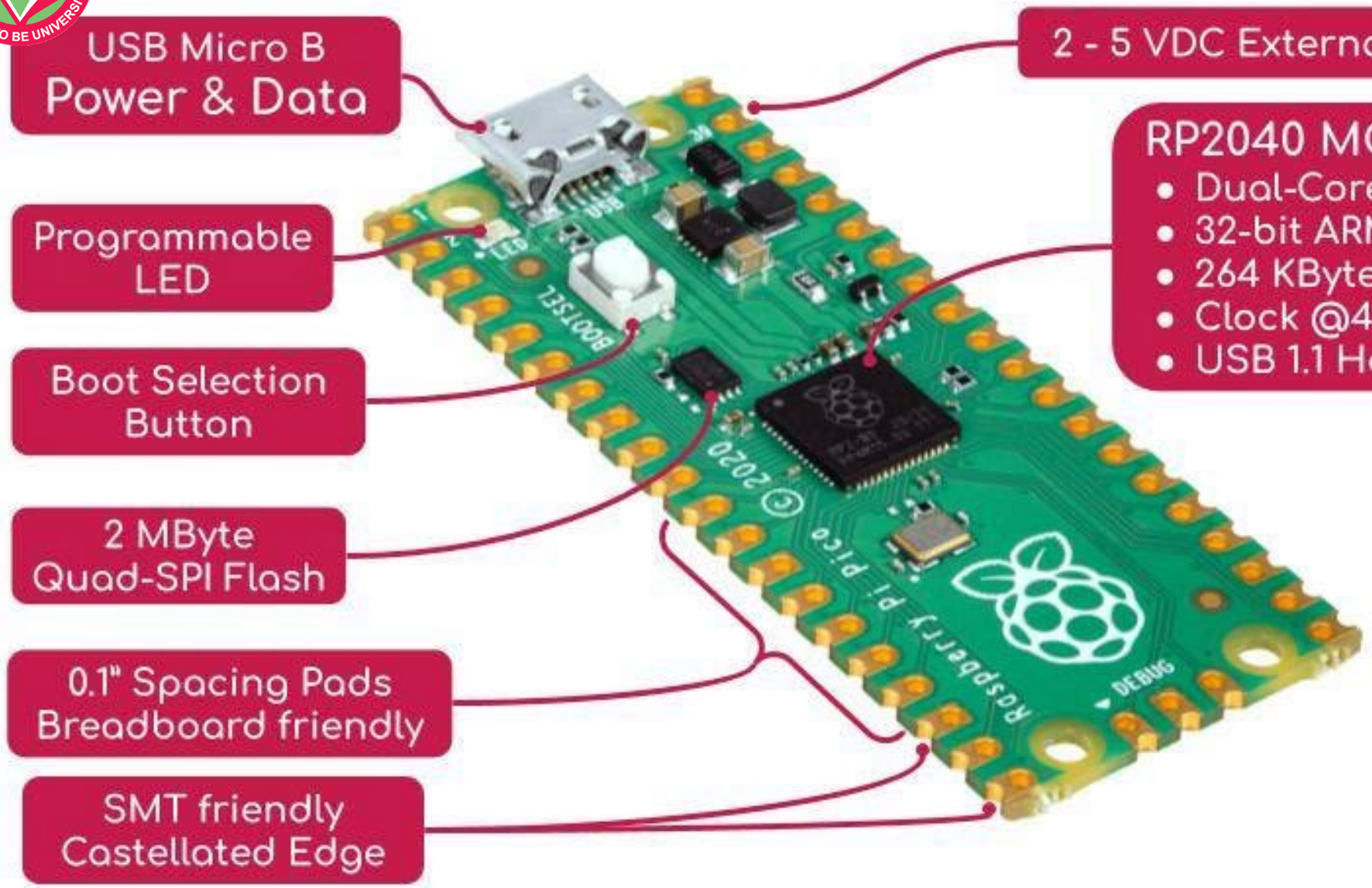
ARDUINO PINS	SWD Pins		
PHYSICAL PIN	POSITIVE SUPPLY	UART1 Pins	UART0 Pins
RESET/ENABLE	GROUND SUPPLY	I2C1 Pins	I2C0 Pins
GPIO PORT/PIN	ANALOG PIN	SPI1 Pins	SPI0 Pins

- **GP29/ADC3** is used to measure VSYS.
- **GP25** is used by debug LED.
- **GP24** is used for VBUS sense.
- **GP23** is connected to SMPS Power Save pin.
- All GPIO pins support PWM. There are total 16 PWM channels.
- All GPIO pins support level and edge interrupts.
- Arduino pins are as per **Arduino-Pico** core by *Earle F. Philhower, III* @earlephilhower
- Arduino's default **Serial** is the USB-CDC of Pico.

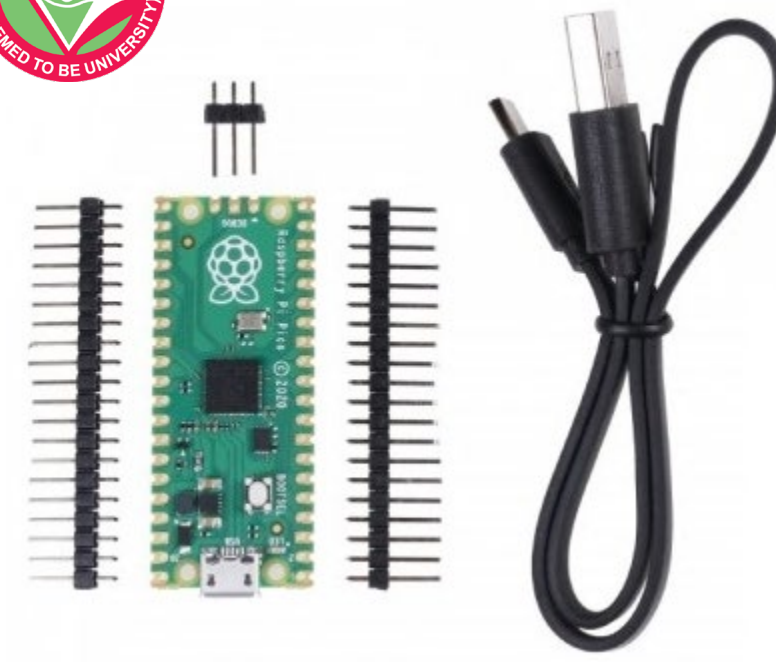
Rev. 0.2, 15-07-2022

Design: Vishnu Mohanan

Raspberry Pi Pico – In Short

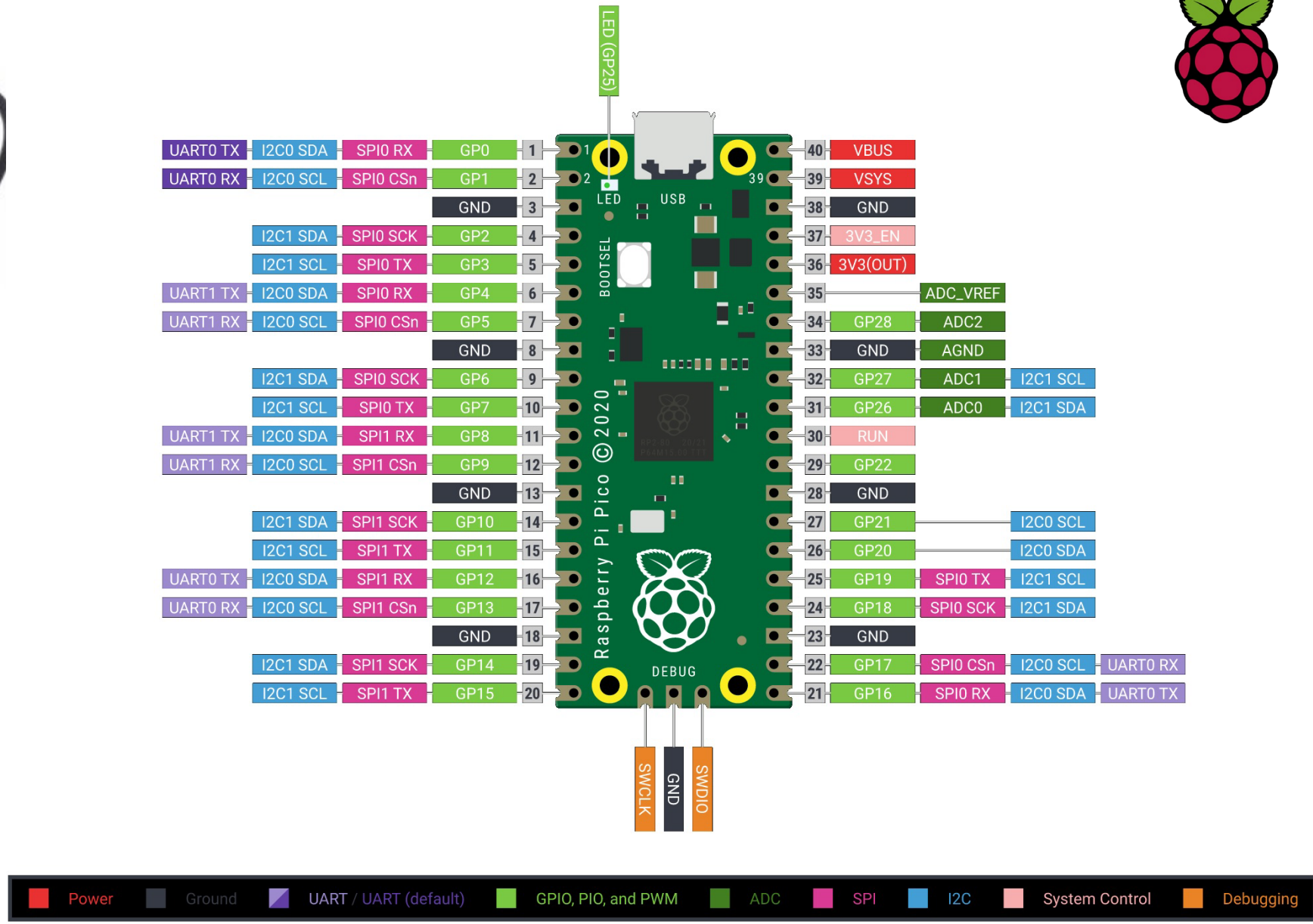


Pi Foundation has released an RP2040 Microprocessor based development board, in the same form factor as an Arduino Nano.



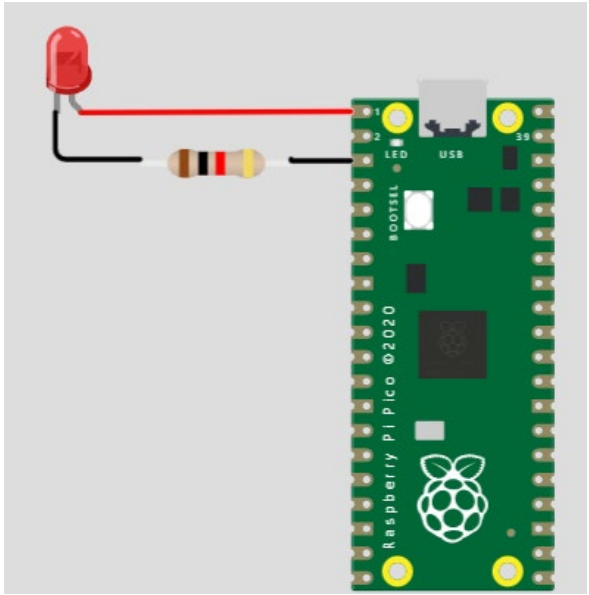
Package Includes:

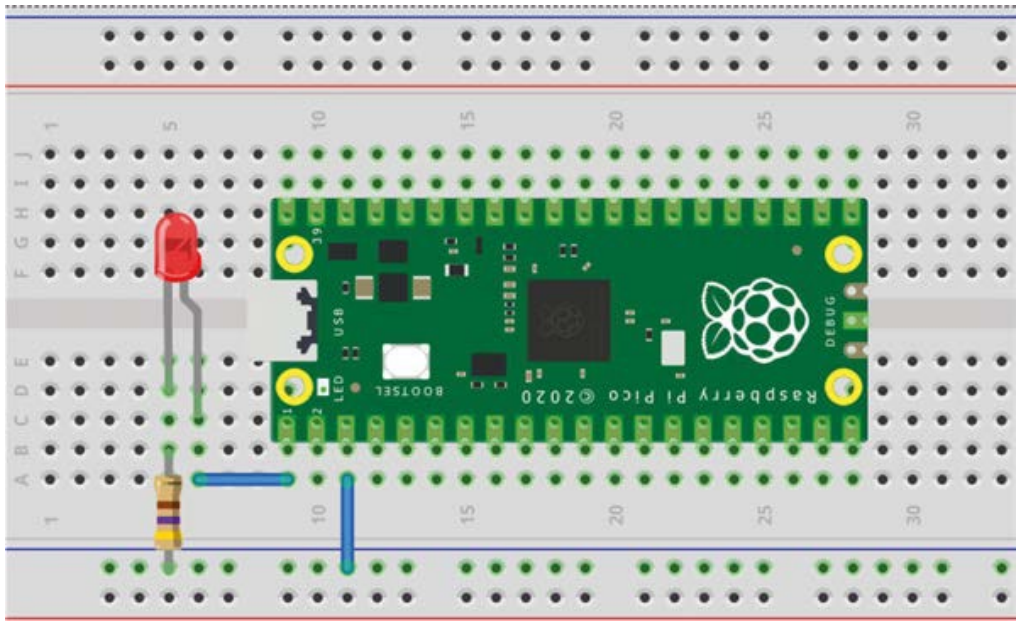
- 1 x Raspberry Pi Pico
- 1 x Micro-USB cable
- 2 x 20 Pin Header
- 1 x 3 Pin Header



Flashing LED – using a timer

- ✓ An external LED is connected to port pin GP0 of the Pico. In this objective a timer is used to flash the LED every 500 ms.
- ✓ A timer is initialized which calls function **Flash_LED** twice (**freq = 2.0**) a second in a periodic manner.
- ✓ LED is flashed using the **toggle** function.





Flashing LED – using a timer

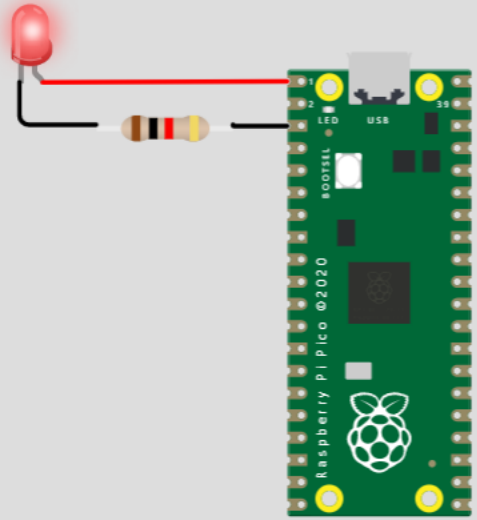
WOKWI SAVE SHARE Docs B

main.py ● diagram.json ● PIO

```
1  from machine import Pin, Timer
2  LED = Pin(0, Pin.OUT)
3
4  tim = Timer()
5  def Flash_LED(timer):
6      global LED
7      LED.toggle()
8  tim.init(freq = 2.0, mode = Timer.PERIODIC, callback = Flash_LED)
```

Simulation

00:13.696 90%

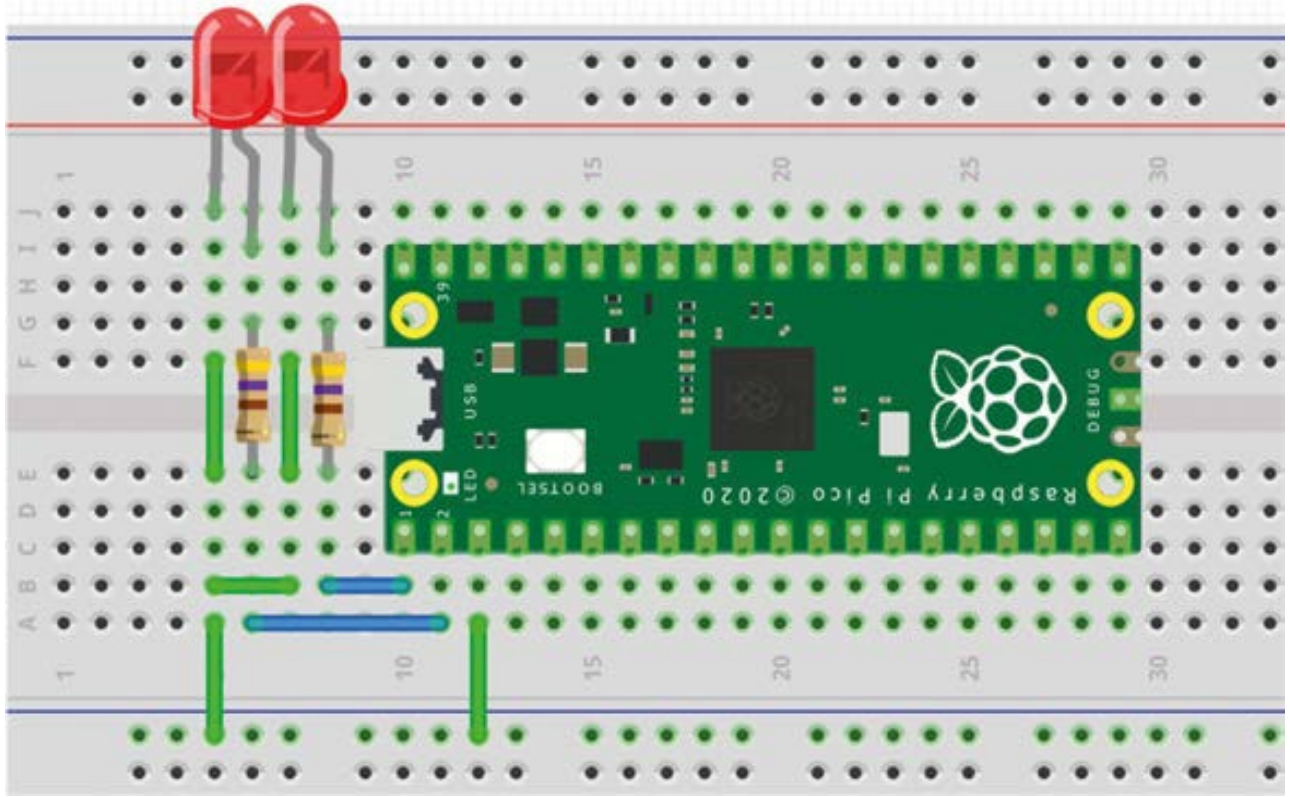


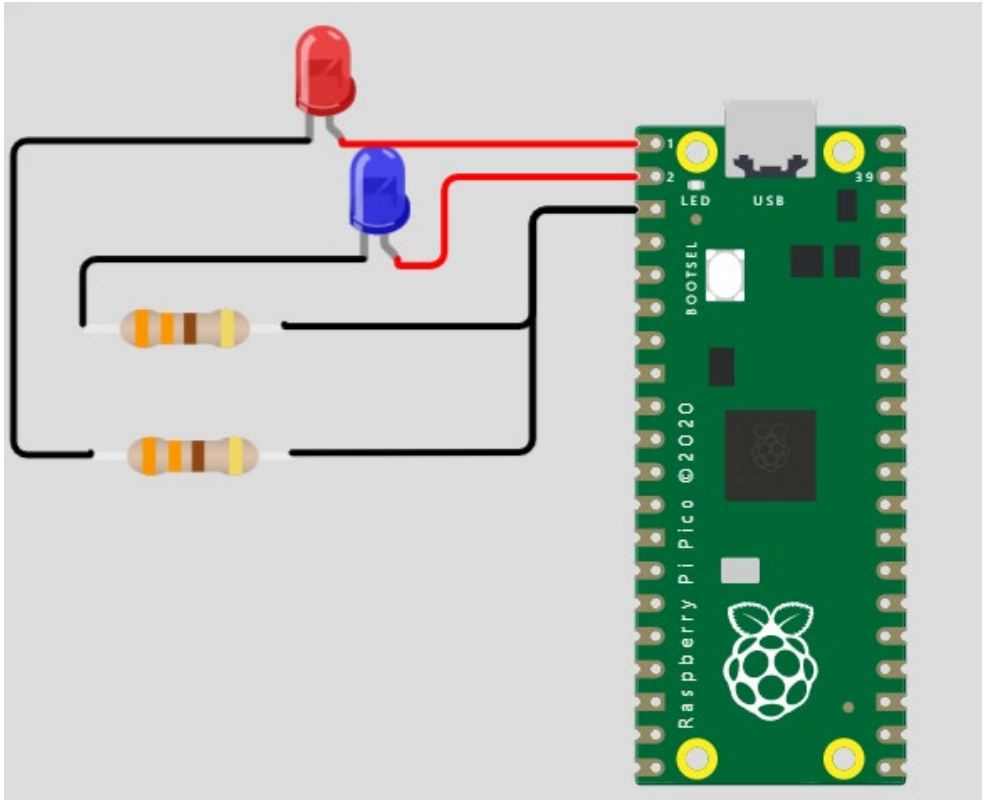
```
MPY: soft reboot
MicroPython v1.18 on 2022-01-17; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>>
```

Alternately flashing LEDs

Two LEDs are connected to the Pico to pins GP0 (pin 1) and GP1 (pin 2).

The LEDs flash alternately every 500 ms.



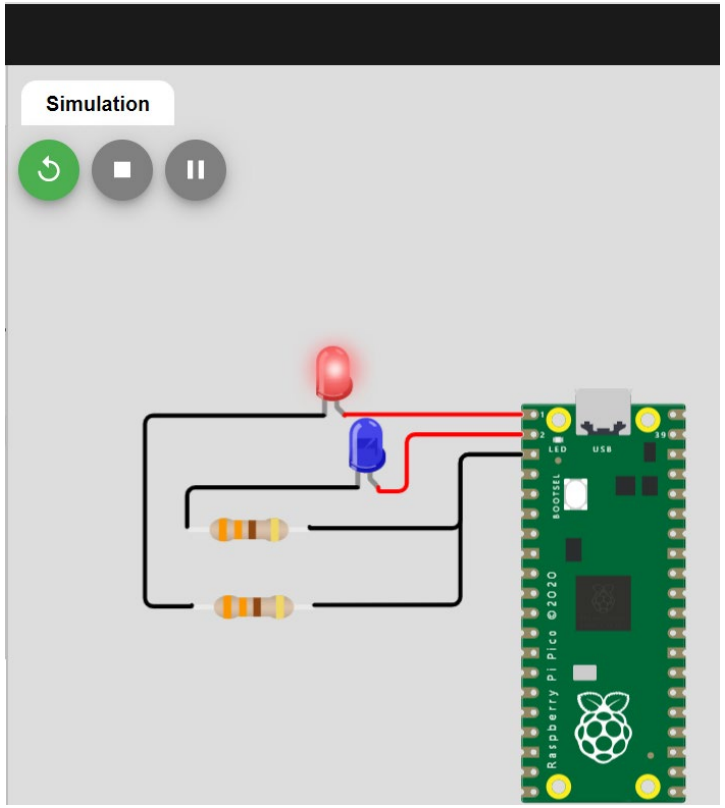


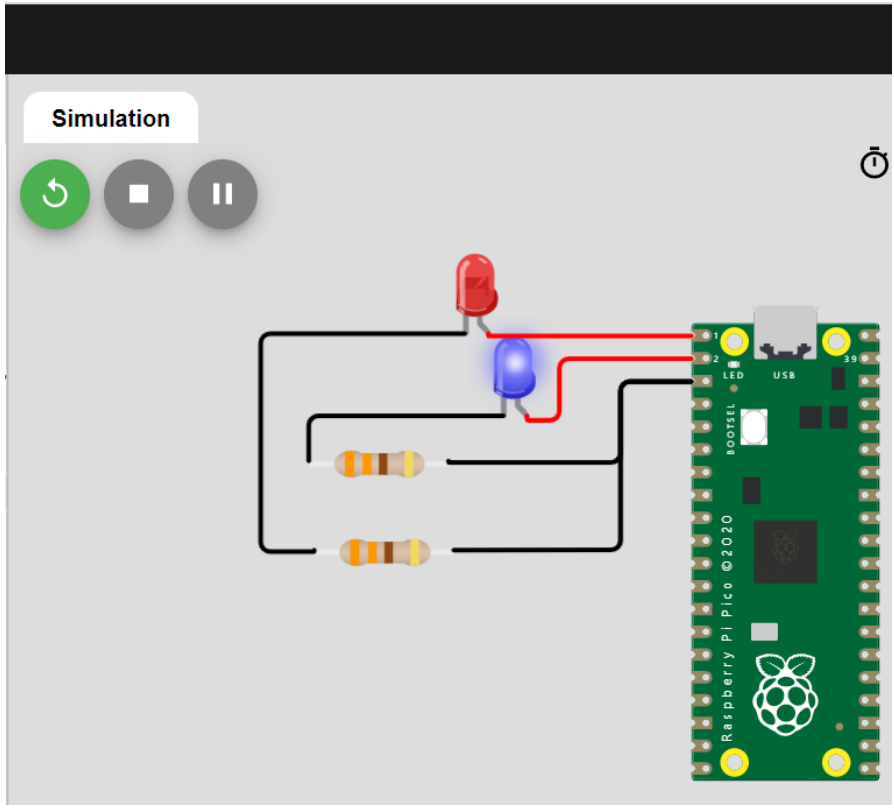
Alternately flashing LEDs

WOKWI SAVE SHARE

main.py ● diagram.json ● PIO

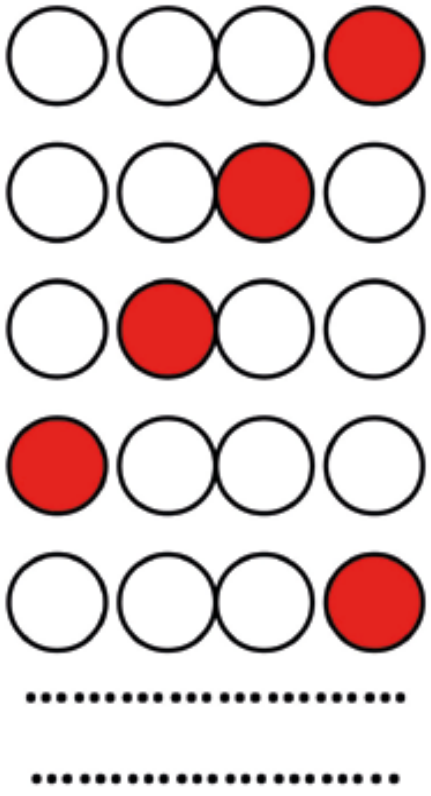
```
1   from machine import Pin
2   import utime
3   LED1 = Pin(0, Pin.OUT)
4   LED2 = Pin(1, Pin.OUT)
5   while True:
6     LED1.value(1)
7     LED2.value(0)
8     utime.sleep(0.5)
9     LED1.value(0)
10    LED2.value(1)
11    utime.sleep(0.5)
12
```

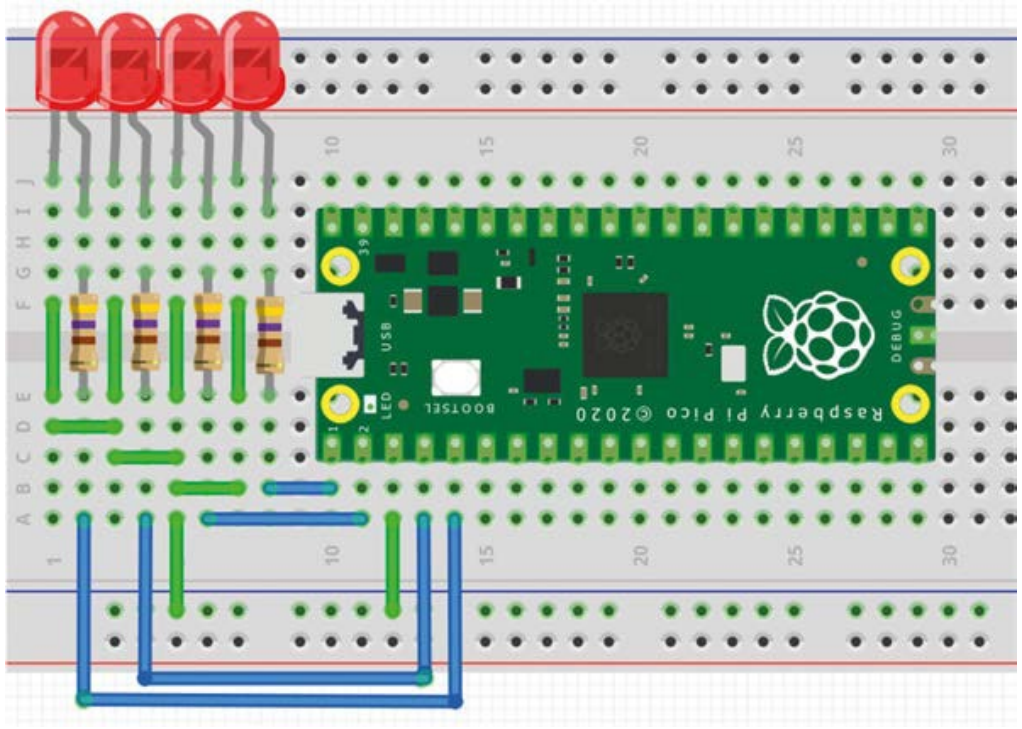


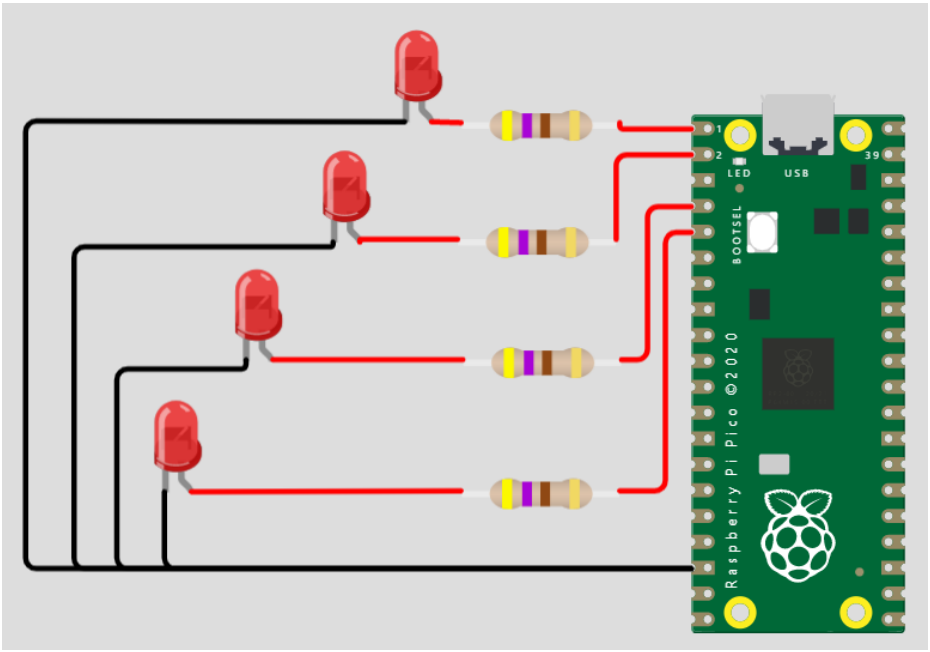


ROTATING LEDs

4 LEDs are connected from port pin GP0 to GP3 of the Pico. The LEDs display pattern of rotating to the left







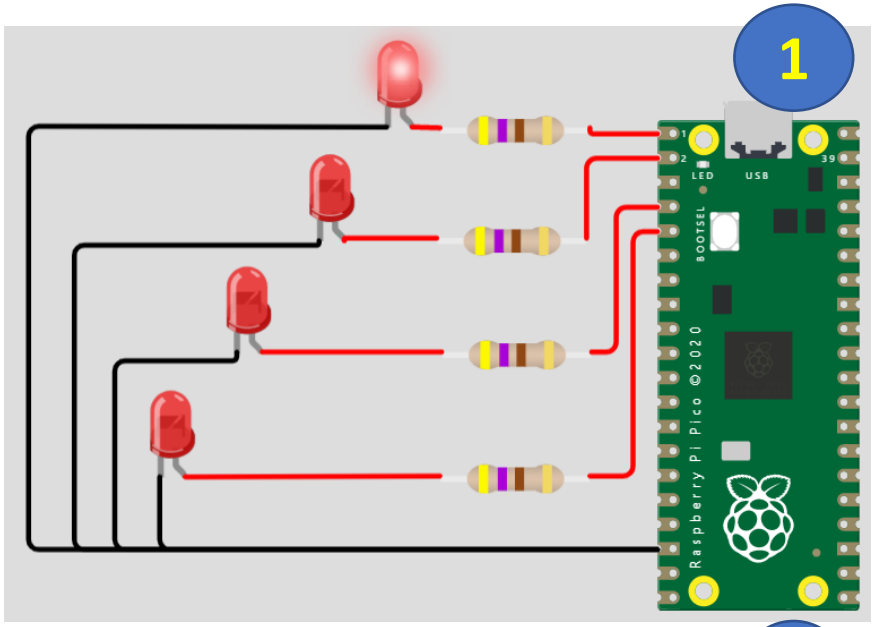
ROTATING LEDs

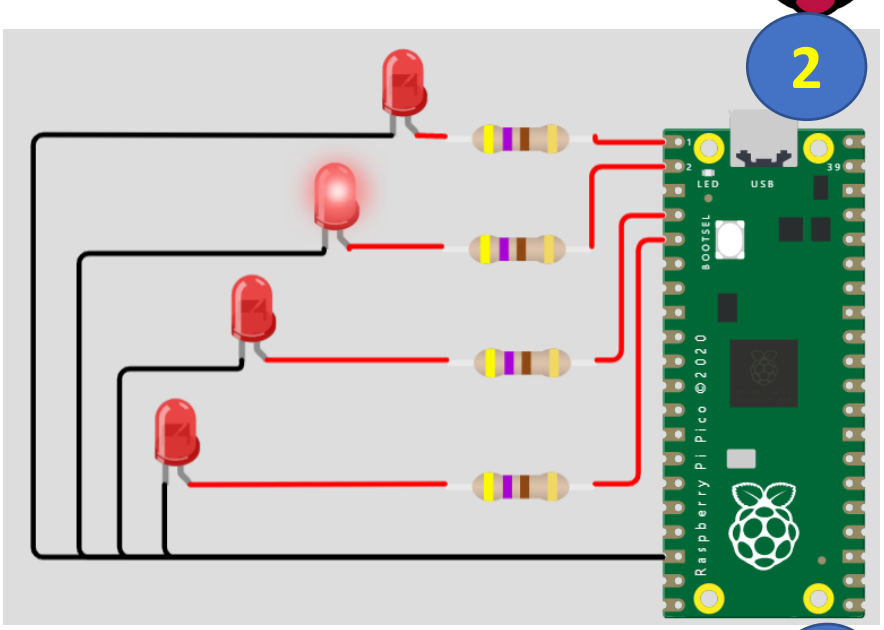
WOKWI SAVE SHARE

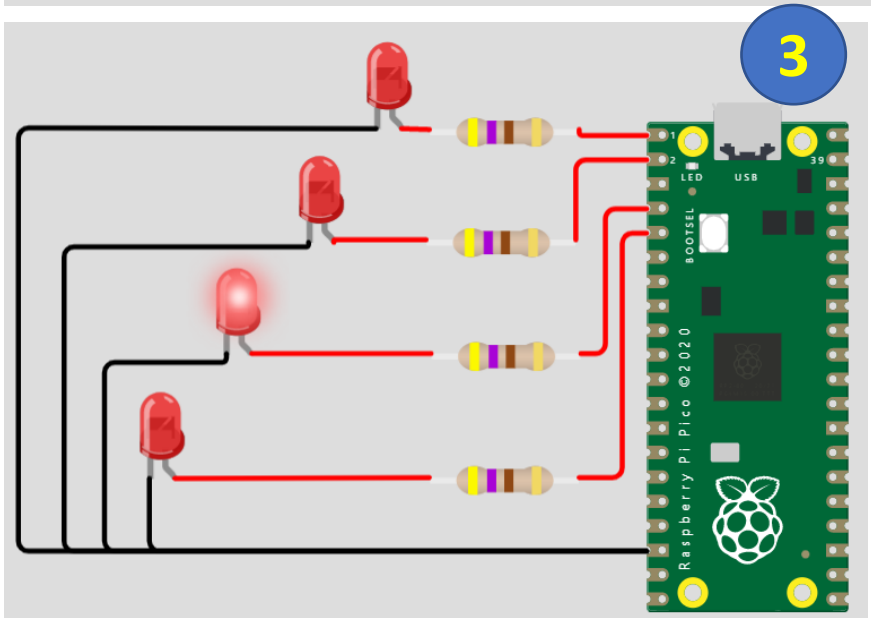
main.py ● diagram.json ● PIO

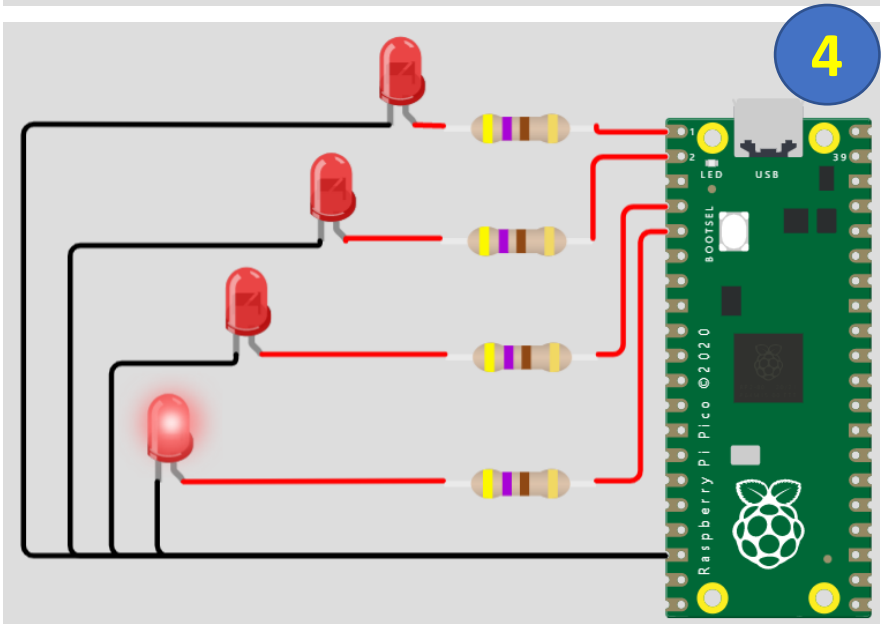
```
from machine import Pin
import utime
LED1 = Pin(0, Pin.OUT)
LED2 = Pin(1, Pin.OUT)
LED3 = Pin(2, Pin.OUT)
LED4 = Pin(3, Pin.OUT)

while True:
    LED1.value(1)
    utime.sleep(0.5)
    LED1.value(0)
    LED2.value(1)
    utime.sleep(0.5)
    LED2.value(0)
    LED3.value(1)
    utime.sleep(0.5)
    LED3.value(0)
    LED4.value(1)
    utime.sleep(0.5)
    LED4.value(0)
```

Can you write a more efficient program??







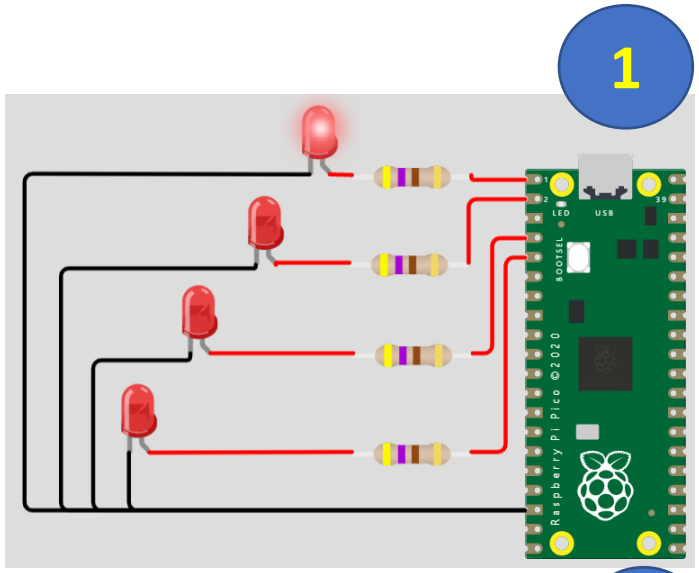


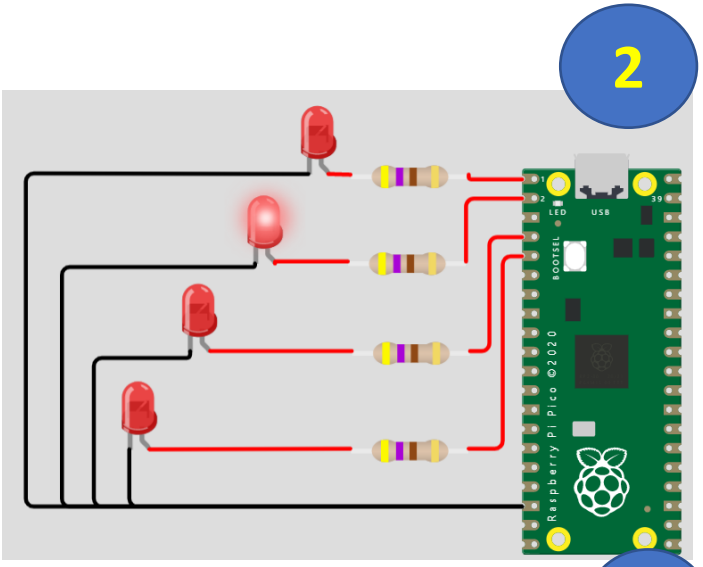
EFFICIENT ROTATING LEDs

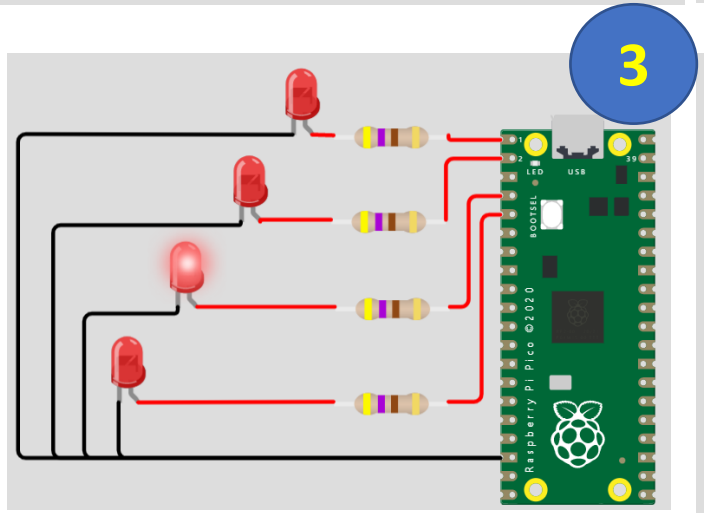
WOKWi SAVE SHARE

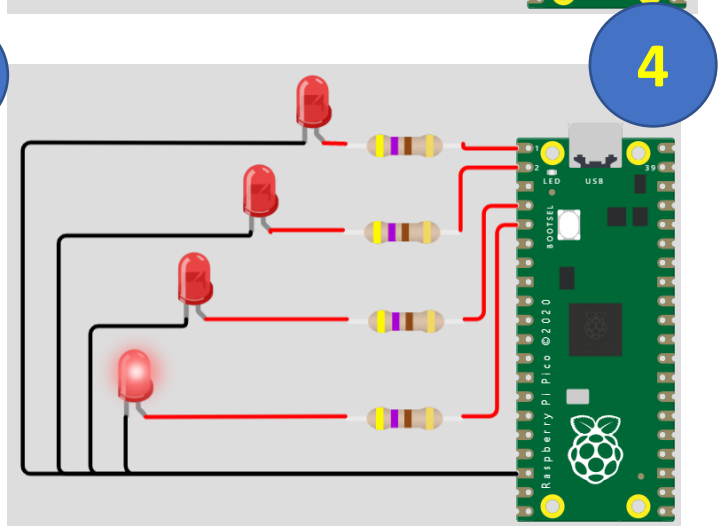
main.py ● diagram.json ●

```
1   from machine import Pin
2   import utime
3
4   LEDS = [0, 1, 2, 3]              # LED ports
5   L = [0, 0, 0, 0]
6
7   for i in range(4):               # Do for all LEDs
8     L[i] = Pin(LEDS[i], Pin.OUT)   # All are outputs
9
10  while True:                      # Do forever
11    for i in range(4):
12      L[i].value(1)                # LED ON
13      utime.sleep(0.5)             # Wait 0.5 second
14      L[i].value(0)                # LED OFF
```





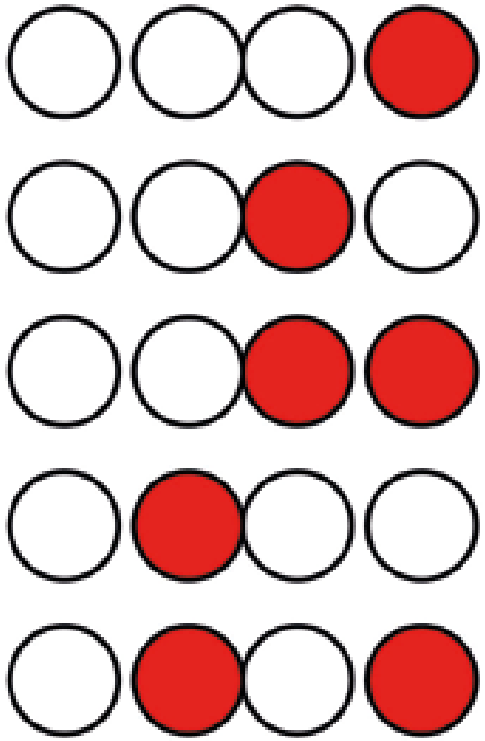


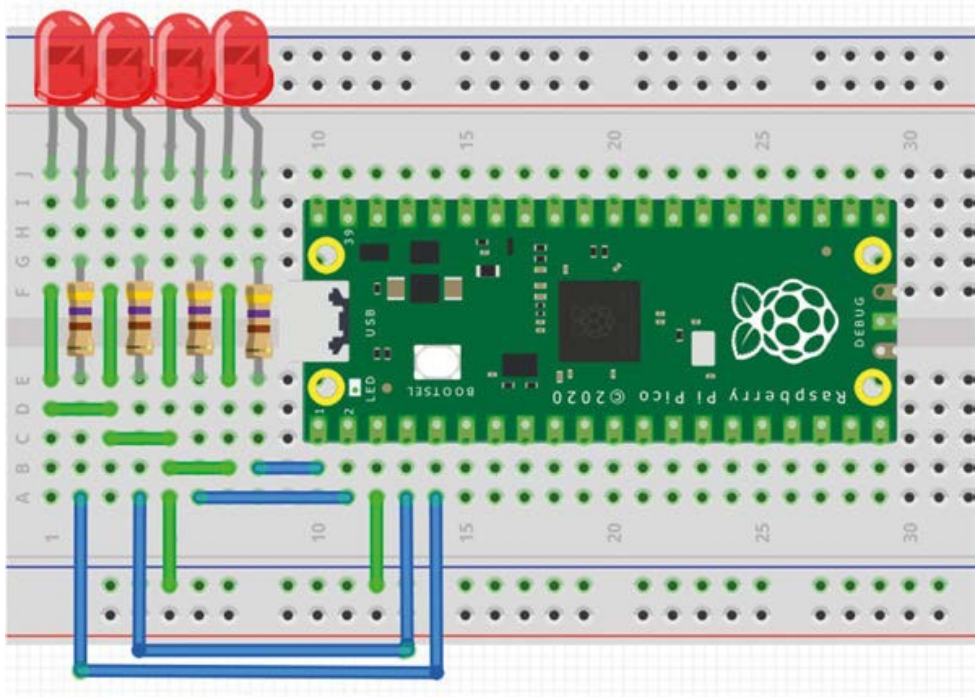


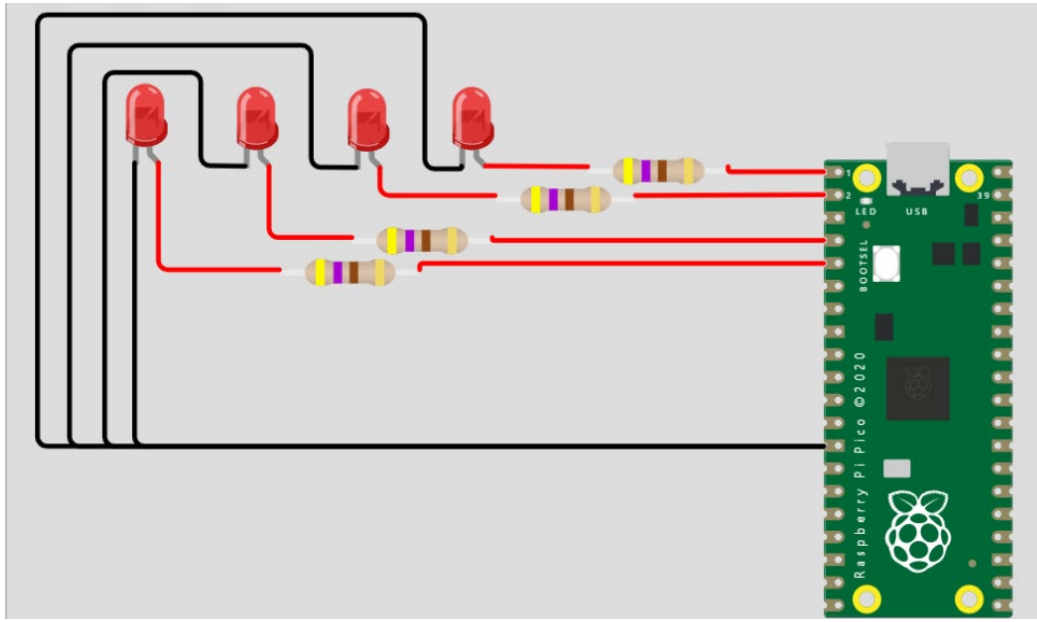
This is especially true if there are more than 4 LEDs.

4-bit Binary-counting LEDs

- ✓ 4 LEDs are connected to Pico.
- ✓ The LEDs count up in binary every second from 0 to 15.





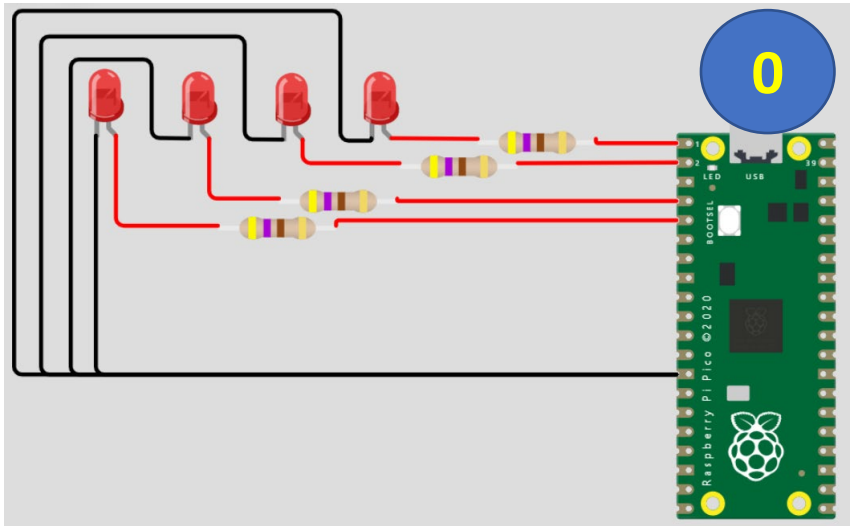


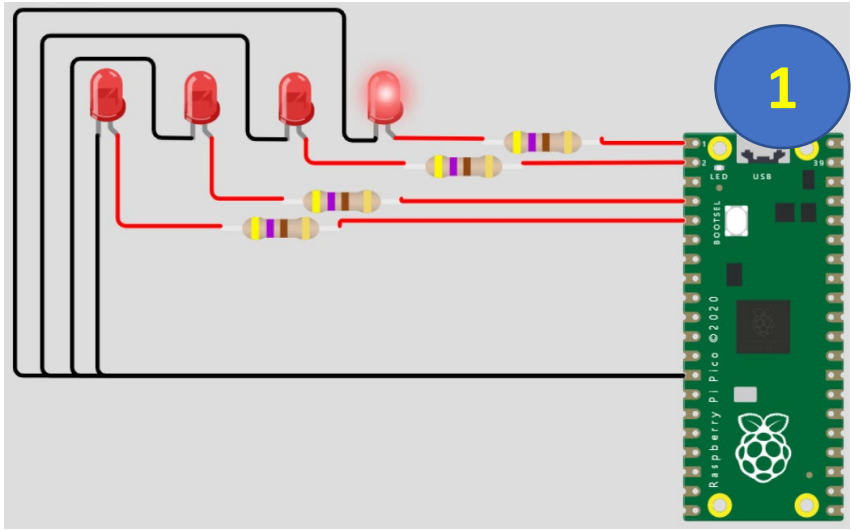
4-bit Binary-counting LEDs

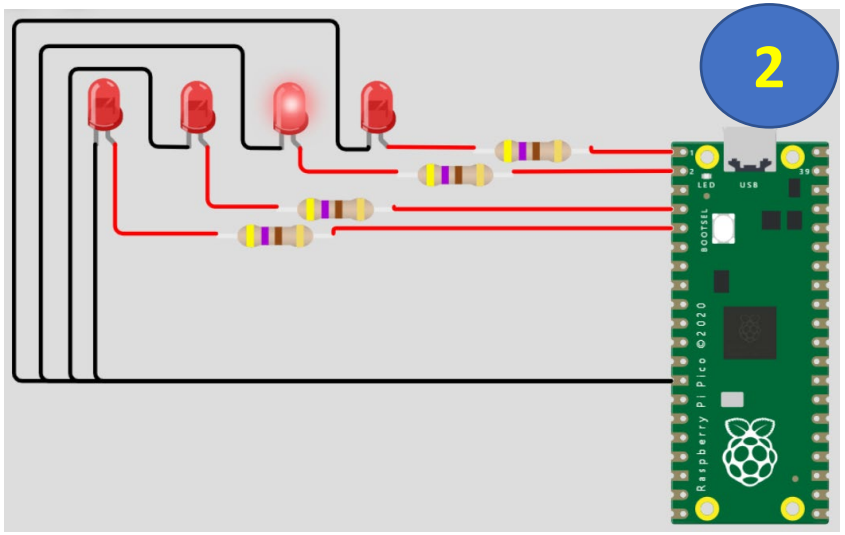
WOKWi SAVE SH

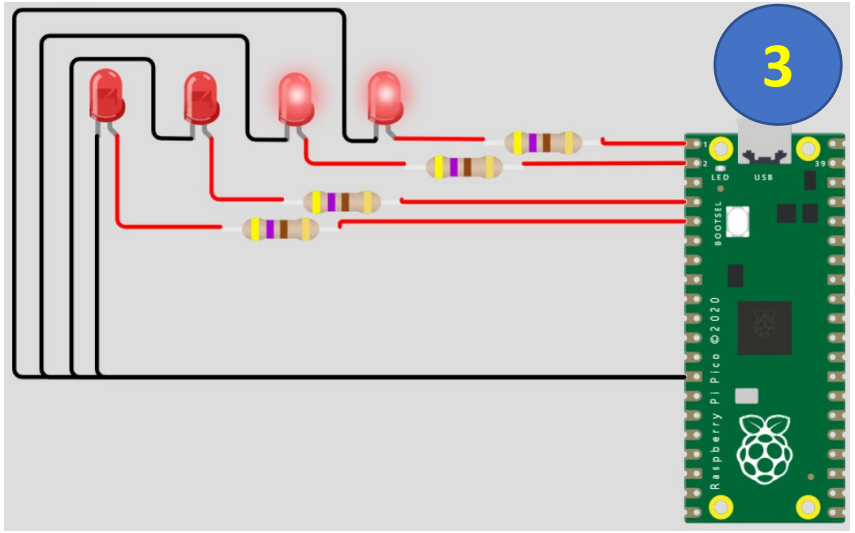
main.py diagram.json

```
1  from machine import Pin
2  from utime import sleep
3
4  led1 = Pin(0, Pin.OUT)
5  led2 = Pin(1, Pin.OUT)
6  led3 = Pin(2, Pin.OUT)
7  led4 = Pin(3, Pin.OUT)
8
9  while True:
10   led4.value(0)
11   led3.value(0)
12   led2.value(0)
13   led1.value(0)
14   sleep(1)
15
16   led4.value(0)
17   led3.value(0)
18   led2.value(0)
19   led1.value(1)
20   sleep(1)
21
22   led4.value(0)
23   led3.value(0)
24   led2.value(1)
25   led1.value(0)
26   sleep(1)
```









Can you write a more efficient program??

and so on.....

Efficient 4-bit Binary-counting LEDs

WOKWi SAVE SHARE

main.py ● diagram.json

```
1   from machine import Pin
2   import utime
3
4   PORT = [3, 2, 1, 0]                # port connections
5   DIR = ["0","0","0","0"]            # port directons
6   L = [0]*4
7   #
8   # This function configures the port pins as outputs ("O") or
9   # as inputs ("I")
10  #
11  def Configure_Port():
12    for i in range(0, 4):
13      if DIR[i] == "0":
14        L[i] = Pin(PORT[i], Pin.OUT)
15      else:
16        L[i] = Pin(PORT[i], Pin.IN)
17    return
18  #
19  # This function sends 4-bit data (0 to 15) to the PORT
20  #
21  def Port_Output(x):
22    b = bin(x)                       # convert into binary
23    b = b.replace("0b", "")          # remove leading "0b"
24    diff = 4 - len(b)                # find the length
25    for i in range (0, diff):
26      b = "0" + b                    # insert leading os
27
28    for i in range (0, 4):
29      if b[i] == "1":
```

WOKWi SAVE SHARE

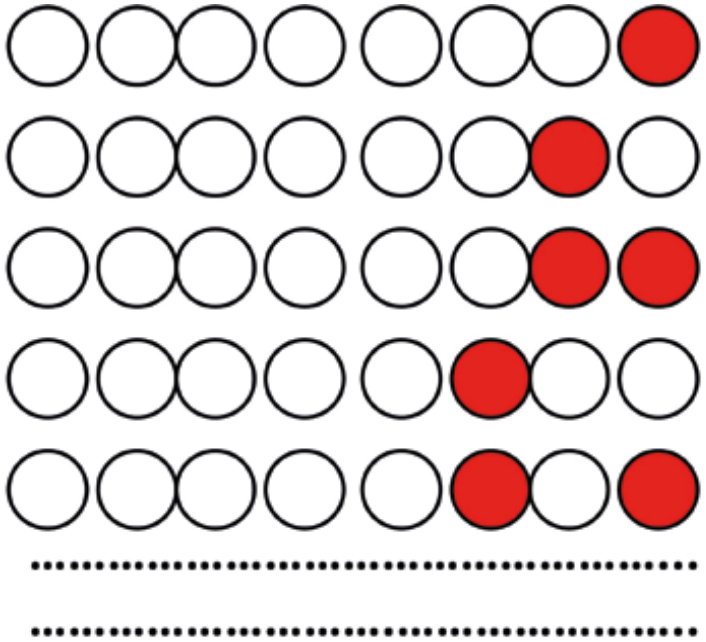
main.py ● diagram.json

```
30        L[i].value(1)
31      else:
32        L[i].value(0)
33    return
34
35  # Configure PORT to all outputs
36  #
37  Configure_Port()
38  # Main program loop. Count up in binary every second
39  #
40  cnt = 0
41  while True:
42    Port_Output(cnt)                 # send cnt to port
43    utime.sleep(1)                   # wait 1 second
44    cnt = cnt + 1                    # increment cnt
45    if cnt > 15:
46      cnt = 0
```

and so on.....

8-bit Binary-counting LEDs

- ✓ 8 LEDs are connected to Pico.
- ✓ The LEDs count up in binary every second from 0 to 255.



Try to implement

Fancy Diwali/Christmas lights

- ✓ 8 LEDs are connected to Pico.
- ✓ The LEDs flash randomly every 250 milliseconds just like Diwali Light or fancy Christmas lights.

Clue:

Generate random numbers between 1 and 255 and then shows how to use these numbers to turn the individual LEDs ON and OFF randomly.

Try to implement by yourself

Fancy Diwali/Christmas lights

WOKWi SAVE SHARE

main.py ● diagram.json ●

```
from machine import Pin
import utime import sleep
import random

PORT = [7, 6, 5, 4, 3, 2, 1, 0]
DIR = ["0","0","0","0","0","0","0","0"]
L = [0]*8

def Configure_Port():
  for i in range(0, 8):
    if DIR[i] == "0":
      L[i] = Pin(PORT[i], Pin.OUT)
    else:
      L[i] = Pin(PORT[i], Pin.IN)
  return

def Port_Output(x):
  b = bin(x)
  b = b.replace("0b", "")
  diff = 8 - len(b)
  for i in range (0, diff):
    b = "0" + b

for i in range (0, 8):
  if b[i] == "1":
    L[i].value(1)
```

WOKWi SAVE SHARE

main.py ● diagram.json ●

```
  else:
    L[i].value(0)
return

Configure_Port()

while True:
  numbr = random.randint(1, 255)
  Port_Output(numbr)
  sleep(0.25)
```