

## 24. Acceleration & Light Sensors

### Introduction:

The Raspberry Pi-Pico has four 12-bit analog inputs - three of which are connected to the I/O pins on the Pi-Pico Breadboard Kit. With these A/D inputs, you can directly read sensors whose output is voltage. With some circuitry, you can also measure sensors whose output is a resistance or a current.

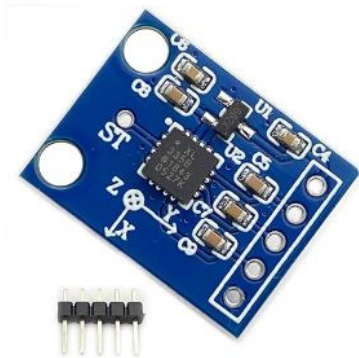
This lecture looks at measuring acceleration and light, With these, we can


- Build a Magic 8-Ball,
- Measuring your Vertical Leap, and
- Build a solar tracker (electronic sunflower)

### Analog Acceleration (ADXL335)


The ADXL335 is an accelerometer with

- 3-5V operation,
- Operable over a full +/- 30g range
- With three analog outputs: Acceleration in the X, Y, and Z direction



 Kiro&Seeu ADXL335 3-Axis Accelerometer Angular Transducer Sensor Module Analog Output Compatible with Ar-duino, (DXL335-MD-1P)  
Brand: Kiro&Seeu  
5.0 ★★★★★ 2 ratings | Search this page

\$12<sup>99</sup>

 prime Two-Day

**Coupon:**  Apply 5% coupon Shop items > | Terms

Save up to 5% with business pricing. Sign up for a free Amazon Business account

Brand	Kiro&Seeu
Material	Copper
Style	Digital
Measuring Range	±30 g
UPC	767217089739

- 0G = 0.7V
- 0.15V per G

Scaling for m/s<sup>2</sup>:

- offset = 0.7V = 200 / 65535
- Digital Acceleration

```
from machine import ADC
from time import sleep_ms

a2d0 = ADC(0)
a2d1 = ADC(1)
a2d2 = ADC(2)

def Read_ADXL335():
    k = 200 / 65535
    x1 = a2d0.read_u16()
    y1 = a2d1.read_u16()
    z1 = a2d2.read_u16()

    Ax = k * (x1 - 13843)
    Ay = k * (y1 - 13779)
    Az = k * (z1 - 13843) - 3.6
    return([Ax, Ay, Az])

while(1):
    [Ax, Ay, Az] = Read_ADXL335()
    Acc = (Ax**2 + Ay**2 + Az**2) ** 0.5

    print('{: 10.3f}'.format(Ax), '{: 10.3f}'.format(Ay), '{:
10.3f}'.format(Az), '{: 10.3f}'.format(Acc))
    sleep_ms(200)
3
3
3
3
`
```

Shell


X	Y	Z	Mag
0.146	0.000	9.556	9.557
-0.098	0.000	9.605	9.605
0.000	-0.146	9.458	9.459

## GY521 Digital Accelerometer

<https://peppe8o.com/using-gyroscope-and-accelerometer-with-mpu6050-raspberry-pi-pico-and-micropython/>




Roll over image to zoom in

 GY-521 MPU-6050 MPU6050 3-Axis Accelerometer Gyroscope Sensor Module 6-axis Accelerometer Gyroscope Sensor Module IIC I2C 3-5V for Arduino 5pcs  
Visit the Teyleten Robot Store  
4.0 ★★★★★ 15 ratings | Search this page

---

\$1.88 (\$2.58 / Item)

 **prime** One-Day  
FREE Returns

<b>Brand</b>	Teyleten Robot
<b>Item dimensions L x W x H</b>	0.79 x 0.59 x 0.12 inches
<b>Material</b>	Copper
<b>Style</b>	Modern
<b>Mounting Type</b>	Flange Mount

---

**About this item**

- Power supply :3-5v (internal low dropout regulator)
- Communication modes: standard IIC communications protocol
- Chip built-in 16bit AD converter, 16-bit data output
- Gyroscopes range: +/- 250 500 1000 2000 degree/sec
- Acceleration range: +2 ±4 ±8 ±16g, Pin pitch 2.54mm

GY-521 digital accelerometer from Amazon

The nice thing about analog sensors is they're pretty easy to use: connect them to power and ground and you get an analog output. The nice thing about digital sensors is they tend to be less noisy (shorter leads) and they don't use up the analog input pins. The challenge with them is coming up with the driver routines.

The GY-521 accelerometer is a digital accelerometer, capable of

- 3V to 5V operation (meaning it will work with 3.3V)
- +/- 2g up to +/- 16g accelerations
- +/- 250 to +/- 2000 degrees per second rotation
- Both I2C and SPI communications.

In terms of hardware and software. Pepe80.com has a very good tutorial on using the GY-521 with a Pi-Pico, along with software and schematics using an I2C interface. In terms of wiring, the connections are:

- Vcc = 3.3V
- GND = 0V
- SCL = GP01
- SDA = GP00

In terms of software, once the driver libraries are loaded onto your Pi-Pico

- imu.py
- vector3d.py

you're ready to read the acceleration and rotational rates (see following code). The execution time is

- 1.96ms to read the acceleration in the XYZ direction
- 1.96ms to read the gyro (rotational velocities) about the XYZ axis

```
# Blog: https://peppe8o.com
# Date: Aug 25th, 2021
# Version: 1.0

from imu import MPU6050
import time
from machine import Pin, I2C

i2c = I2C(0, sda=Pin(0), scl=Pin(1), freq=400000)
imu = MPU6050(i2c)

while True:
    # Following print shows original data get from library. You can
    # uncomment to see raw data
    #print(imu.accel.xyz, imu.gyro.xyz, imu.temperature, end='\r')

    # Following rows round values get for a more pretty print:
    ax=round(imu.accel.x, 2)
    ay=round(imu.accel.y, 2)
    az=round(imu.accel.z, 2)
    gx=round(imu.gyro.x)
    gy=round(imu.gyro.y)
    gz=round(imu.gyro.z)
    tem=round(imu.temperature, 2)
    print(ax, "\t", ay, "\t", az, "\t", gx, "\t", gy, "\t", gz, "\t", tem, "
", end="\r")

    # Following sleep statement makes values enough stable to be
    # seen and
    # read by a human from shell
    time.sleep(0.2)
```

shell

```
x"      y"      z"
0.19  0.0  0.98  -2  0  -2  28.41
```

Main routine for reading a GY-521 accelerometer

<https://peppe8o.com/using-gyroscope-and-accelerometer-with-mpu6050-raspberry-pi-pico-and-micropython/>

## Magic 8-Ball



Roll over image to zoom in



### Mystic 8 Ball Decision Making Fortune Telling Ball Retro Game Novelty Black Eight Ball

Brand: MystiCabin

4.5 ★★★★★ 113 ratings | Search this page  
800+ bought in past month

g88

prime One-Day

FREE Returns

Save up to 3% with business pricing. Sign up for a free Amazon Business account  
Earn 5% back (\$0.49 in rewards) on the amount charged to your Prime Visa.

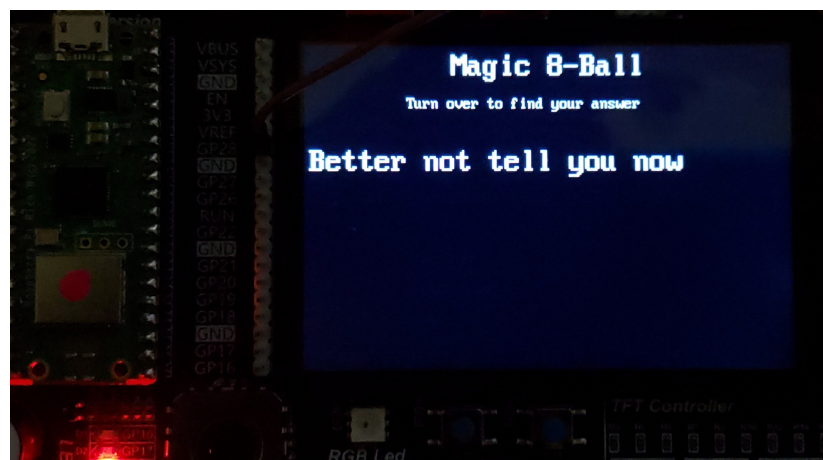
- HOW TO PLAY: When asking the Mystic Black Ball a question, just turn the ball and find the answer revealed in the small window. For all other question, shake the ball. Turn the toy upside-down and look inside the window on the bottom - this is where your secret answers, the future appears!
- It is the novelty toy, that lets anyone seek advice about their future! A fun game and a classic novelty toy for adults, Gather some friends, take turns shaking the ball, and watch the hilarity ensue.
- The fortune teller ball will be a great gift idea for friends, family members, know what the future has in store with a shake of this Mystic Black Ball, makes a memorable treat.
- 20 Possible answers like "ABSOLUTELY, VERY LIKELY, YOU CAN COUNT ON IT, FOCUS AND ASK AGAIN..." please get the mystic ball to find the answer that you want.

Once you can measure acceleration, there are lots of things you can do. One example is to build a Magic 8-Ball. This is device which can answer any of life's difficult questions with one of 20 random results. To operate the Magic 8-Ball,

- Shake the Magic 8-Ball
- Ask it a question, then
- Turn it over to see what the answer is.

The following code uses the analog accelerometer to measure the acceleration in the z-direction.

- Start with the z-axis pointing down
- Ask a question
- Shake the accelerometer up and down briskly three times, then
- Turn it over to get your answer.



Result of Magic 8-Ball Code. When you shake then turn over the acceleration sensor, your question is answered

```
0from machine import ADC
from time import sleep
from random import randrange

Fortune = [ 'It is certain',
            'Reply hazy, try again',
            'Dont count on it',
            'It is decidedly so',
            'Ask again later',
            'My reply is no',
            'Without a doubt',
            'Better not tell you now',
            'My sources say no',
            'Yes definitely',
            'Cannot predict now',
            'Outlook not so good',
            'You may rely on it',
            'Concentrate and ask again',
            'Very doubtful',
            'As I see it, yes',
            'Most likely',
            'Outlook good',
            'Yes',
            'Signs point to yes']

def Read_ADXL335():
    k = 200 / 65535
    x1 = a2d0.read_u16()
    y1 = a2d1.read_u16()
    z1 = a2d2.read_u16()

    Ax = k * (x1 - 13843)
    Ay = k * (y1 - 13779)
    Az = k * (z1 - 13843) - 3.6
    return([Ax, Ay, Az])

[Ax, Ay, Az] = Read_ADXL335()

print('Shake for a reading')

while(1):
    print('-----')
    # Shake three times for a reading
    for i in range(0,3):
        #print('Shake #',i)
        while(Az < 20):
            [Ax, Ay, Az] = Read_ADXL335()
        while(Az > 5):
            [Ax, Ay, Az] = Read_ADXL335()
    #print('Your Fortune')
    N = randrange(20)
    print(Fortune[N])
    sleep(1)
```

Results:

```

-----
MPY: soft reboot
Shake for a reading
-----
It is decidedly so
-----
Without a doubt
-----
As I see it, yes
-----

```

### Measuring your Vertical Leap

Another thing you can do with an accelerometer is measure your vertical leap.

- When at rest, you should have an acceleration of 9.8 m/s<sup>2</sup> due to gravity
- When in free-fall, the acceleration you feel should drop to zero m/s<sup>2</sup>

By measuring how long you are experiencing zero g's, you should be able to measure the duration of your jump, and then calculate your height as

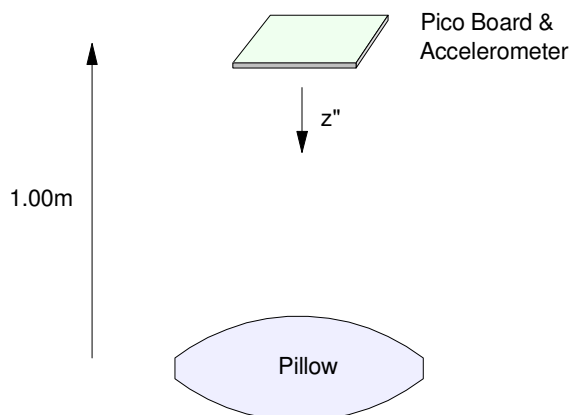
$$d = \frac{1}{2}at^2$$

where t is the time from your apogee to the ground, or

$$d = \frac{1}{2}a\left(\frac{t}{2}\right)^2$$

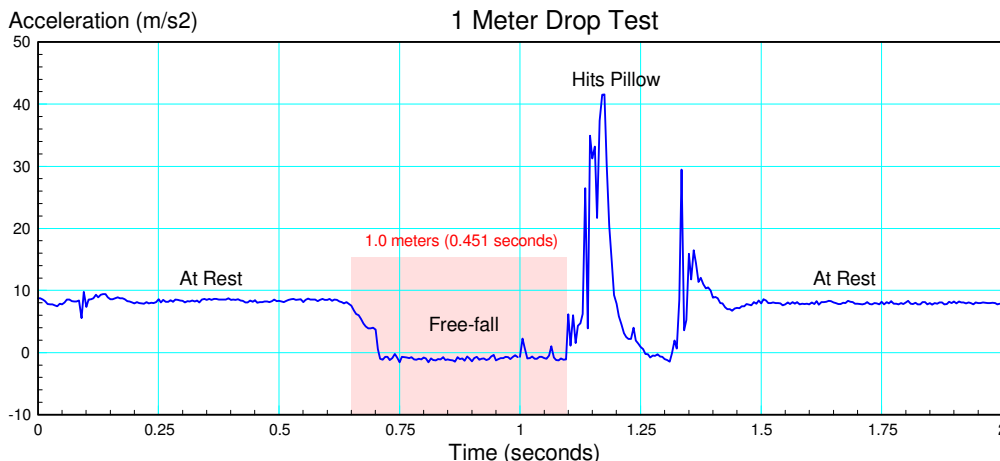
where t is the total time of your jump.

**Free-Fall Test:** Start out by using the code for measuring your heart beat and replace the sensor with the ADXL335 analog accelerometer (a digital one would work too.) To check if the code is working, drop the sensor from a height of 1.00 meter onto a pillow and record the acceleration as it falls.



Free-Fall Test: The acceleration should read as zero while falling

The acceleration was measured for 2.00 seconds with a sampling rate of 5ms for this drop tes. The resulting acceleration in the z-direction was as follows:



Free Fall Test. Acceleration in the z-direction is recorded every 5ms

Note:

- When in free-fall, the recorded acceleration is about zero (there's a slight calibration error)

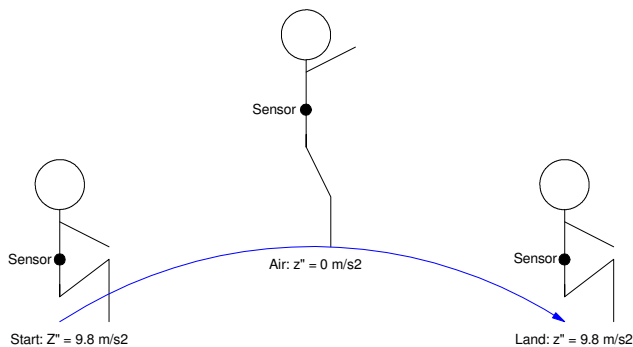
If you take the time to be when the acceleration is less than 2 m/s<sup>2</sup>,

- The duration of the free-fall event is  $t = 0.400$  seconds
- Which corresponds to a distance of 0.784 meters

For the height to read 1.00 seconds, the time of free-fall should be 0.451 seconds. This would be time time from when the acceleration drops below 6 m/s<sup>2</sup> and ends when the acceleration goes above +1 m/s<sup>2</sup>. So, this set up sort of works.

**Jump Test:** Now that I'm somewhat confident I can measure the time when an object is in free-fall, repeat this experiment with measuring the acceleration of a person (me) when jumping.

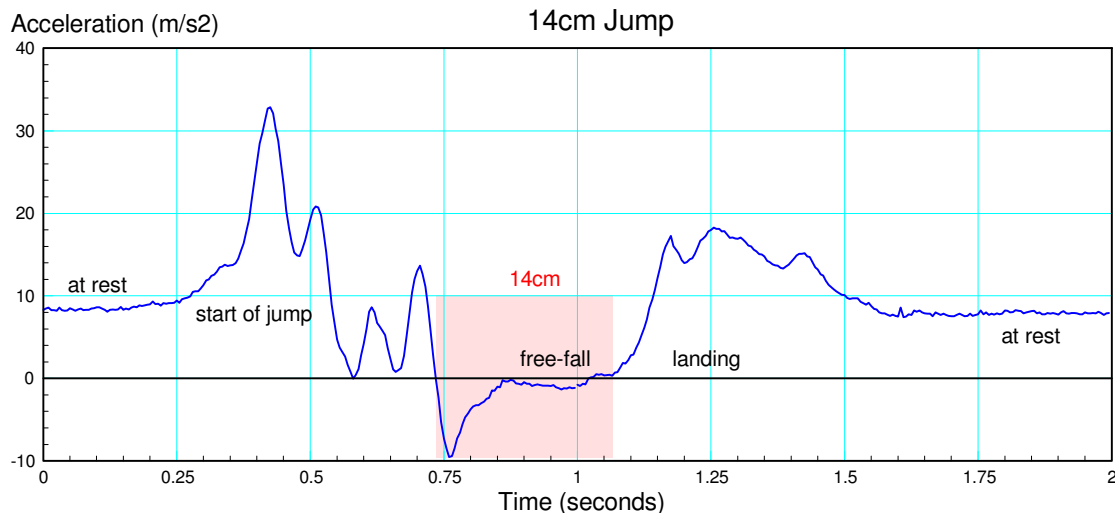
- At the start of the experiment, I'll be crouching down, ready to jump.
- When ready, start recording data (press GP15 button), then
- Jump



Jump Test: Measure your acceleration in the z-direction when jumping



The acceleration in the z-direction will then be recorded for 2 seconds with a sampling rate of 5ms (same as before). The resulting acceleration vs. time is in the following figure.



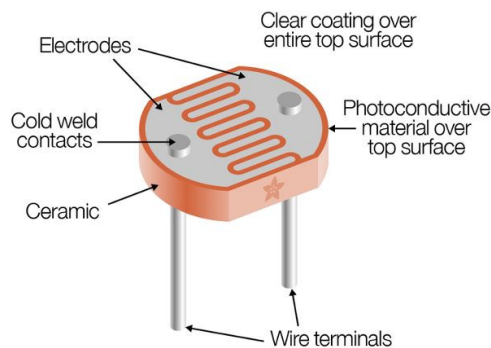
Measured acceleration in the z-direction for a 14cm jump

From observation, it looks like my vertical leap is about 14cm (not real impressive). From the data, the time when the acceleration is less than +1 m/s<sup>2</sup> is:

- Start of jump: 0.735 seconds
- End of jump: 1.080 second
- Duration = 345ms
- Height = 14.58 cm

which seems about right. With a little more coding, the time of flight can be measured and displayed automatically.

## CdS Light Sensors



PDV-P8001 CdS Light Sensor (Adafruit.com)

Going back to the thermistor discussion, a thermistor is a sensor whose resistance changes with temperature. If you replace the thermistor with a sensor whose resistance changes with light levels, you get a light sensor.

Cadmium Sulfide is a chemical whose resistance changes with light level. This property is used to make CdS light sensors. In general, the resistance - lux relationship is of the form:

$$R = a \cdot \left(\frac{1}{Lux}\right)^b$$

Given two data points, you can find a and b.

Take for example a PDV-P8001 CdS photoresistor from Adafruit. From the data sheets,

- At 10 Lux (1 foot candle), the resistance is between 3k and 11k (assume 7k nominal)
- The sensitivity is 0.6 (b in the above equation)

This allows you to solve for {a, b}

$$R = a \cdot Lux^{-0.6}$$

At 10 Lux

$$7000 = a \cdot (10)^{-0.6}$$

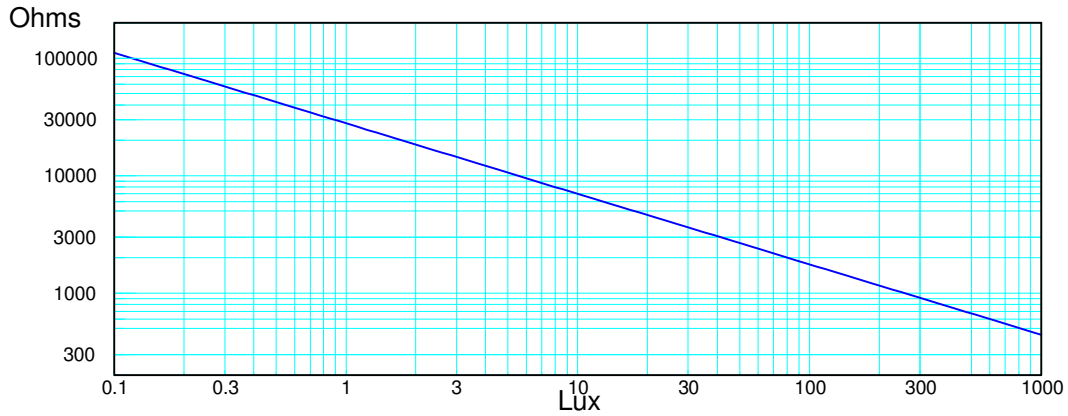
$$a = 27,867\Omega$$

meaning

$$R \approx 27,867 \cdot (Lux)^{-0.6}$$

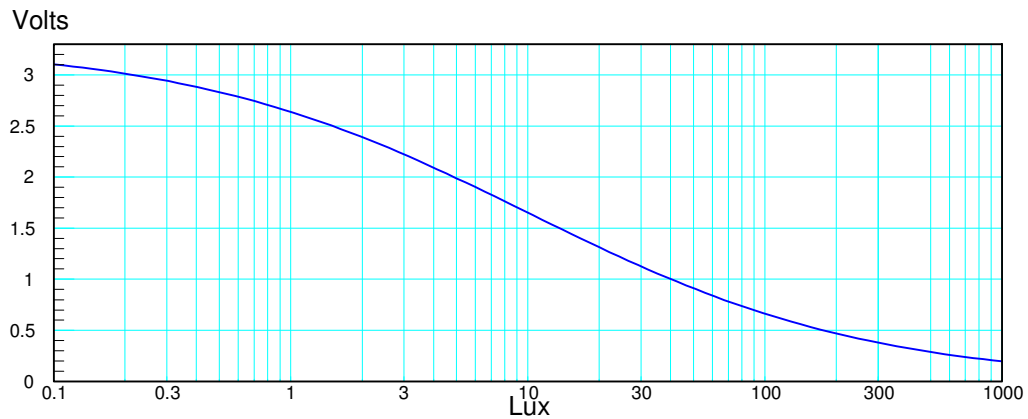
or the light-resistance relationship should be:

```
>> Lux = logspace(-1, 3, 100)';
>> R = 27867 * Lux.^ -0.6;
>> loglog(Lux, R)
>> xlabel('Lux')
>> ylabel('Ohms')
```



Resistance vs. Light Level for a P8001 CdS Light Sensor

Using a voltage divider, this can be converted to a voltage a Pi-Pico can read. With such a wide range of resistance, however, it will be difficult covering the whole range with just a single divider. For example, if you use a 10k resistor for the divider (mid-band resistance approximately), the resulting voltage will be as follows. Note that the slope of the curve tapers off at the edges. This implies a loss of sensitivity as you move away from 10 Lux.



Voltage vs. Light Level for a CdS light sensor

## PhotoVoltaic (PV) Light Sensors

Another way to measure light is to use a photovoltaic solar cell. These are essentially solar panels which convert sunlight into electricity.



Typical photovoltaic sensor from Amazon.

Sunlight has a nominal energy density of 1370 W/m<sup>2</sup> (NASA). Solar cells convert this to electricity with efficiencies of 13% to 23% for commercial solar cells, up to 39% for research grade solar cells (NREL). You can measure this energy density by placing a resistor across the solar cell and measuring the energy produced.

For example, using a 5cm x 10cm solar cell pulled out of a dead battery, the energy produced can be measured as:

Condition	Voltage	Power	Energy Density
July 14, 2024		47 Ohm resistor	
Dawn (8am)	2.08V	92.05mW	48%
Noon	3.00V	191mW	100%
Evening: 6pm	2.758V	161.8mW	84.7%
Dusk: 9pm	0.15V	0.47mW	0.2%

Once you can measure light intensity, you can

- Detect the light level in a room (light on or off)
- Detect if a refrigerator door is open or closed
- Move a pan and tilt servo motor until it is pointing directly at the sun (solar tracker)

## Summary

Acceleration sensors detect motion. These are available with both analog and digital outputs. With an acceleration sensor, you can detect motion, shaking, orientation, etc.

Light sensors let you measure the light intensity. These let your program respond to light levels (room light on or off, daytime or night time.)

Both are fairly easy to interface to a Pi-Pico.

---

## References

### Pi-Pico and MicroPython

- [https://github.com/geekpi/pico\\_breakboard\\_kit](https://github.com/geekpi/pico_breakboard_kit)
- [https://micropython.org/download/RPI\\_PICO/](https://micropython.org/download/RPI_PICO/)
- <https://learn.pimoroni.com/article/getting-started-with-pico>
- <https://www.w3schools.com/python/default.asp>
- <https://docs.micropython.org/en/latest/pyboard/tutorial/index.html>
- <https://docs.micropython.org/en/latest/library/index.html>
- <https://www.fredscave.com/02-about.html>

### Pi-Pico Breadboard Kit

- <https://wiki.52pi.com/index.php?title=EP-0172>

### Other

- <https://docs.sunfounder.com/projects/sensorkit-v2-pi/en/latest/>
- <https://electrocredible.com/raspberry-pi-pico-external-interrupts-button-micropython/>
- <https://peppe8o.com/adding-external-modules-to-micropython-with-raspberry-pi-pico/>
- <https://randomnerdtutorials.com/projects-raspberry-pi-pico/>
- <https://randomnerdtutorials.com/projects-esp32-esp8266-micropython/>